SunATM™ SBus Adapters Manual

Sun

The Network Is the Computer™

Sun Microsystems Computer Company
2550 Garcia Avenue
Mountain View, CA 94043 USA
415 960-1300 fax 415 969-9131

Part No.: 802-4439-10
Revision A, January 1996

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.

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.

Please
Recycle

Adobe PostScript™

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

Figures

Tables

Preface

SunATM™ SBus Adapters Manual provides installation instructions for the SunATM-155/MFiber SBus Adapter 2.0 and the SunATM-155/UTP5 SBus Adapter 2.0. This manual also describes how to install and configure the SunATM 2.0 software.

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

How This Book Is Organized

Chapter 1, "Introducing the SunATM-155 SBus Adapters," introduces the SunATM adapters and software.

Chapter 2, "Installing SunATM-155 SBus Adapter," describes how to install and verify the SunATM adapters.

Chapter 3, "Installing SunATM Software," describes how to install the SunATM 2.0 software using the software installation scripts

Chapter 4, "Configuring the SunATM Interface," defines how you can edit the SunATM configuration files to best suit your network.

Chapter 5, "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-155 SBus Cards Specifications," lists the SunATM SBus Adapter specifications.

Appendix C, "Troubleshooting and Error Messages," provides a troubleshooting section for the software installation and configuration, and it 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, "Advanced Configurations," describes how to edit the SunATM configuration files for non-homogeneous networks.

Appendix G, "Running Diagnostic Tests," provides pointers to the SunDiag™ system exerciser and the SunVTS™ test suite, which both test the SunATM SBus adapters.

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 `.login` file. Use `ls -a` to list all files. `machine_name% You have mail.`

`AaBbCc123`	What you type, contrasted with on-screen computer output	`machine_name%` **`su`**
		`Password:`

AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type `rm` *filename*.
AaBbCc123	Book titles, new words or terms, or words to be emphasized	Read Chapter 6 in the *User's Guide*. These are called *class* options. You *must* 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.

In the U.S.A.	Outside the U.S.A.
Fax 1-801-373-6798	Fax 1-801-373-6798
Call 1-800-247-0250	Call 1-801-342-3450
World Wide Web: `http://www.sun.com/sundocs/catalog.html`	

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 the SunATM-155 SBus Adapters 1

The SunATM-155/MFiber SBus Adapter 2.0 and SunATM-155/UTP5 SBus Adapter 2.0 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.

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

SunATM-155 SBus adapters highlights are:

- Conform to IEEE 1496
- Support 155-Mbps operation over:
  - 62.5/125 µ Multimode fiber (SunATM155/MFiber 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

- Support 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 OBP revision level on your system, type **`.version`** at the `<#0> ok` prompt.

Figure 1-1 shows the SunATM-155/MFiber Adapter SBus adapter 2.0 and Figure 1-2 shows the SunATM-155/UTP5 Adapter SBus adapter 2.0.

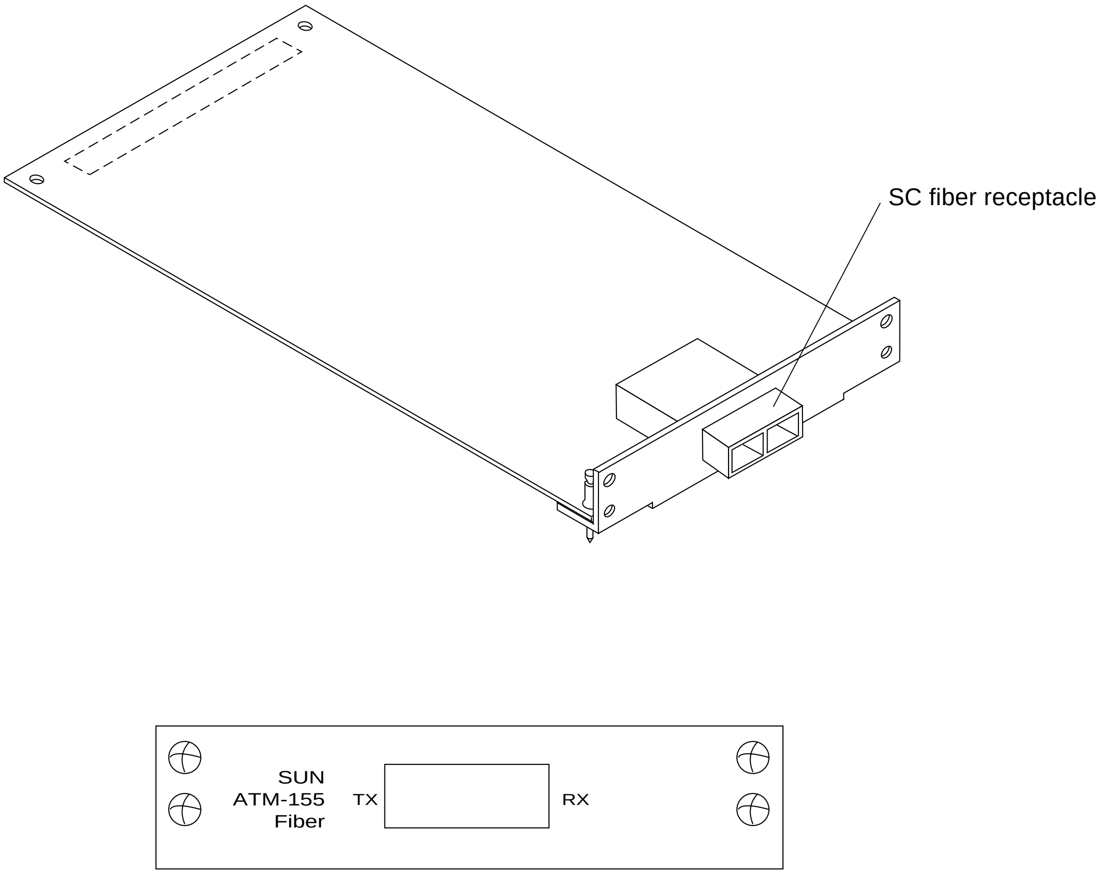


Figure 1-1 SunATM-155/MFiber SBus Adapter 2.0 and Back Panel

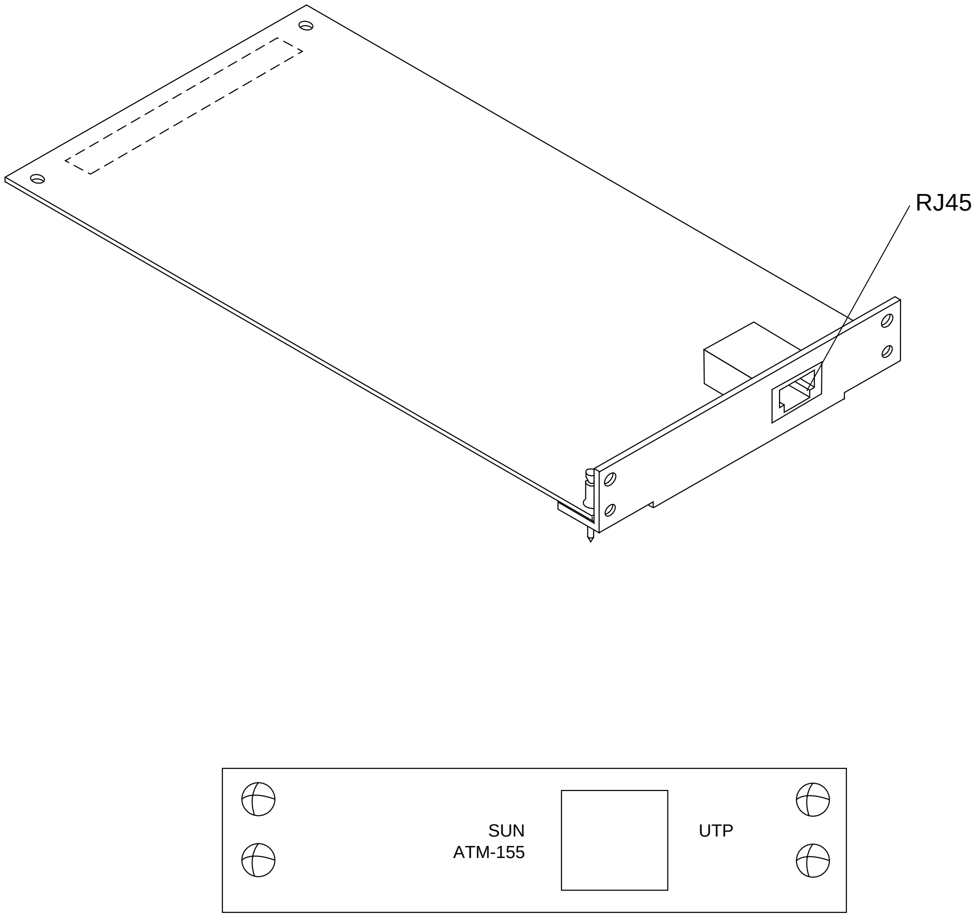


Figure 1-2 SunATM-155/UTP5 SBus Adapter 2.0 and Back Panel

1.1 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/MFiber SBus Adapter 2.0 - Multimode fiber cable with an SC connector
- SunATM-155/UTP5 SBus Adapter 2.0 - Category 5 UTP with a RJ-45 connector

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

Table 1-1 shows the SBus based Sun-4U, Sun-4m, Sun-4d, and Sun-4c architecture systems that support the SunATM-155 SBus adapters 2.0.

Table 1-1 Platform Architecture with Examples of Systems

Platform Architecture	System Type
Sun-4U	Sun Ultra™ 1 Sun Ultra 2
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

1.2 Software Requirements

Note – Install SunATM-155 adapter first. Then install the SunATM 2.0 software and configure the ATM interface.

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

The SunATM 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.

1.3 Network Configuration Information

During the installation of the SunATM software packages, you will be prompted to answer several specific network configuration questions. Before you begin the software installation, you must obtain the information listed in Section 3.1.2, "Compiling Network Configuration Information," on page 3-2.

Note – You will need this information for each SunATM SBus card on your system.

1.4 Differences Between the SunATM 1.0 and SunATM 2.0 Software

Besides supporting local area network (LAN) emulation and the new SunATM-155 SBus adapters, the SunATM 2.0 software has been changed for easier network configuration. This section describes the configuration differences between the SunATM 1.0 and SunATM 2.0 software.

1.4.1 New Configuration Files

In addition to the Classical Internet Protocol (IP) configuration file, `/etc/aarconfig`, there are two new SunATM configuration files in the `/etc` directory: `atmconfig` and `laneconfig`. `atmconfig` provides general configuration information, identifying which software features must be supported on each interface. Information in `/etc/atmconfig` also replaces the `/etc/hostatm.sa`*N* files which existed in the SunATM 1.0 software. `laneconfig` provides configuration information for interfaces that will run LAN Emulation.

1.4.2 `/etc/hostatm.sa`*N* files not required

In the SunATM 1.0 software, the `hostatm.sa`*N* files identifies which interfaces should be initialized and brought up at boot time. A listing of the SunATM interfaces now appears in the `/etc/atmconfig` file, so the additional `hostatm.sa`*N* files are no longer necessary. The new boot script uses the `/etc/atmconfig` file at boot time.

1.4.3 New Boot Script

The configuration of SunATM 1.0 interfaces during the system boot process was handled by the `S79atm` script, which ran while multi-user mode (run level 2) was coming up. In the SunATM 2.0 software, that function is performed by `S60sunatm`. This script still runs as multi-user mode comes up, but earlier than in the SunATM 1.0 software. For system administrators, there is a major result of this change: the script now runs before the network support script (`S69inet`), so SunATM interfaces *will be* included when the system determines if it should come up as a router or not. By default, a system, running the Solaris operating environment, will run as a router if two or more network interfaces are configured, unless the `/etc/rc2.d/S69inet` file is changed (see the `in.routed(1M)` and `rc2(1M)` man pages for more information).

Therefore, under the SunATM 2.0 software, a system with a single ATM interface and on-board ethernet will come up as a router. This was not the case in the SunATM 1.0 software, because the ATM interfaces were not configured until after `S69inet` had run and made the router decision.

In addition, there is now a second script that runs after `S69inet`, `S79sunatm`. The `S79sunatm` script has been added to re-run the IP over the SunATM setup scripts, if necessary. This process allows any IP hostnames appearing in SunATM configuration files to be resolved over non-ATM network information service (NIS) or domain name service (DNS) interfaces, if possible.

1.4.4 `aarpd` *Has Become* `aarsetup`

In the SunATM 1.0 software, there was a daemon `aarpd` which worked with the Classical IP kernel software to provide Classical IP support. In SunATM 2.0, most of the functionality of `aarpd` was moved into the kernel, so the daemon has now been replaced by a setup program, `aarsetup`. `aarsetup` runs when an interface is brought up, then exits. Therefore, when changes are made to `/etc/aarconfig`, the user only needs to run `aarsetup`; there are no daemons to kill and restart or signals to send.

Installing SunATM-155 SBus Adapter 2

Before installing the SunATM 2.0 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, or other compatible operating systems.

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.

If you are only upgrading the SunATM 1.0 software to the new SunATM 2.0 software, and your system has the earlier SunATM-155 SBus (1.0) adapters, the limit of one adapter per SBus still applies.

2.1 SBus Adapter Installation

1. **Wrap the wrist strap around your wrist twice with the conductive adhesive tape against your skin.**

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

2. **Peel the liner from the copper foil at the opposite end of the wrist strap, and attach the copper end of the wrist strap to the metal casing of the power supply in the system unit.**
3. **Remove the SunATM-155 adapter from the antistatic bag, spread the bag on a firm surface to provide a protective mat, and place the SunATM-155 adapter on the bag.**

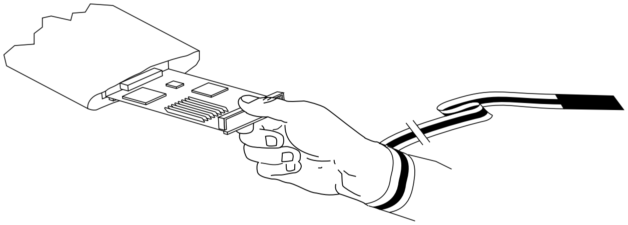


Figure 2-1 Handling the SunATM-155 Adapter

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

4. **Install the SunATM-155 adapter according to the SBus installation procedures in the hardware installation or service manual for your system.**

Note – Do not boot the operating system until SunATM-155 installation is verified. See “Testing the SunATM-155 SBus Adapter Before Booting the System” on page 2-6.

5. **Verify SunATM-155 installation by executing a test command.**
   See "Testing the SunATM-155 SBus Adapter Before Booting the System" on page 2-6.

Sections 2.1.1 and 2.1.2 provide an introduction to the physical connectors and wiring characteristics of the SunATM-155/MFiber and SunATM-155/UTP5 adapters, respectively.

Caution – Do not change the SBus slot in which a SunATM-155 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 adapter and added a second adapter to the system. Refer to the on-line man page about `path_to_inst` for more information.

2.1.1 SunATM-155/MFiber SBus Adapter 2.0

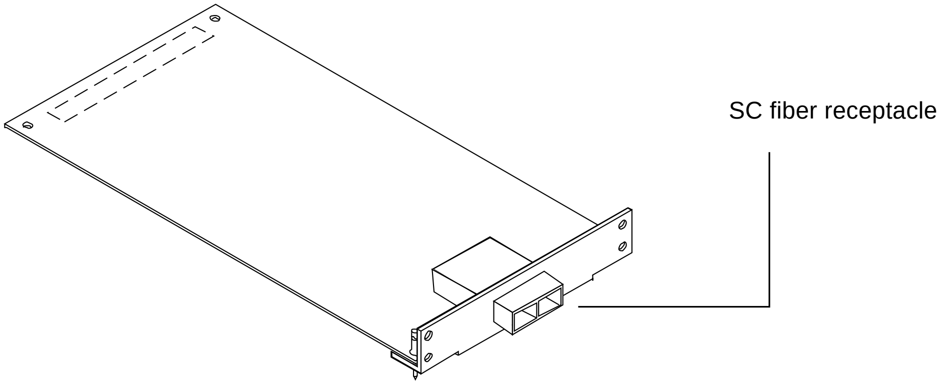


Figure 2-2 SunATM-155/MFiber Adapter 2.0

Extender Plate

A sheet metal back plate adapter kit is included with the SunATM-155 SBus adapter. You must attach this extender plate to the SBus adapter before installing it in some older systems. Refer to the hardware installation or service manual that shipped with your system for information about installing SBus adapters.

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

Wiring Configuration

The SunATM-155/MFiber 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.

Connecting the SBus 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.**

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

2.1.2 SunATM-155/UTP5 SBus Adapter 2.0

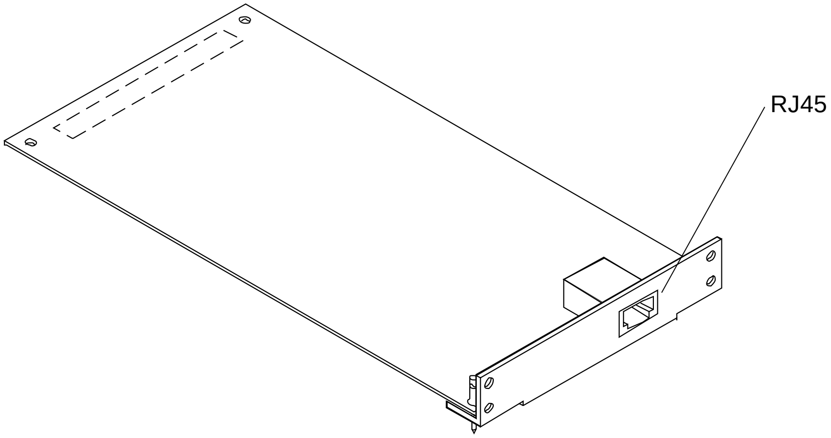


Figure 2-3 SunATM-155/UTP5 Adapter 2.0

Extender Plate

A sheet metal 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.

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").

Connecting the 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.**

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

2.2 Testing the SunATM-155 SBus Adapter Before Booting the System

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

1. **Use `show-devs` to find out SBus adapter 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 2.0 shipped with the SunATM 2.0 software will be identified by the driver name `ba`. Older SunATM-155 1.0 SBus adapters shipped with the SunATM 1.0 software will still be supported by the SunATM 2.0 software, but they will be identified by the driver name `sa`.

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

```
<#0> ok show-devs /iommu/sbus
/iommu@f,e0000000/sbus@f,e0001000/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,f8fffffc
/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/eeprom@0,200000
/obio/zs@0,0
/obio/zs@0,100000
/iommu@f,e0000000/sbus@f,e0001000
/iommu@f,e0000000/sbus@f,e0001000/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
/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 SunATM Software 3

Before installing and configuring the SunATM software, you must install the SunATM SBus adapter first. See Chapter 2, “Installing SunATM-155 SBus Adapter,” for more information.

See the Glossary for descriptions of the ATM and SunATM terms used in this chapter. Chapter 4, “Configuring the SunATM Interface,” and Chapter 5, “Classical IP and LAN Emulation Protocols,” also provide more information about ATM protocols and the SunATM inplementation of these protocols.

If you have any problems with the software installation, see Appendix C, “Troubleshooting and Error Messages,” for helpful tips and pointers.

3.1 Before Installing the SunATM Software

Before installing the SunATM 2.0 software, you must first remove any previous version of the SunATM software from your system, and you must find the specific configuration information of the system.

3.1.1 Removing Older SunATM Software Packages

Before installing the SunATM 2.0 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
```

3.1.2 Compiling Network Configuration Information

During the installation of the SunATM software packages, you will be prompted to answer a number of specific network configuration questions. Before you begin the software installation, you should know the information in Table 3-1.

Note – You will need this information for each SunATM SBus adapter on your system.

Table 3-1 SunATM Network Information Form

ATM Configuration Questions	Answers
What version of the ATM Forum User Network Interface (UNI) Specification does your switch use? (The SunATM software supports either UNI version 3.0 or version 3.1.)	
What type of ATM Support for Internet Protocol (IP) is required for your network? The choices are: Classical IP LAN Emulation Both Classical IP and LAN Emulation No IP Support	
Depending on what kinds of protocols you choose, you will be required to know this additional information:	
For Classical IP, you will be prompted for this information:	
ATM address:	
IP address:	
IP hostname:	
If the system is an ATM ARP client, you will need to know the ATM ARP server's ATM address:	
For LAN Emulation, you will be prompted for this information:	
ATM address:	
IP address:	
IP hostname:	
If the system is an ATM ARP client, you will need to know the ATM ARP server's ATM address:	

3.2 Installing and Configuring the SunATM 2.0 Software

Note – SunATM adapters are supported on systems running the Solaris software environment, revision 2.4 or later (or other compatible operating systems).

You must use one of these methods to install the required driver software using the SunATM 2.0 CD-ROM that came with the SBus adapter.

Note – The SunATM Device Drivers (SUNWatm) and SunATM Runtime Support Software (SUNWatmu) packages are required for SunATM adapters.

3.2.1 Adding the Software Packages Using `pkgadd`

Note – If you have any problems during the SunATM 2.0 software installation, see Section 3.2.2, “Troubleshooting During Software Installation,” and Appendix C, “Troubleshooting and Error Messages,” for more information.

---

1. **Become superuser (root).**
2. **Insert the SunATM 2.0 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_0`
- 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 2.0 software packages.**

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

```
# /usr/sbin/pkgadd -d /cdrom/sunatm_2_0 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
```

During the package installation, you will be asked various questions about your hardware configuration. Depending on your hardware and your network set-up, you will be prompted for the information described in Section 3.1.2, “Compiling Network Configuration Information.”

Note – If you unsure on how to answer a configuration question, type a question mark to receive more information about the question.

Note – The SunATM Device Drivers (SUNWatm) and SunATM Runtime Support Software (SUNWatmu) packages are required for SunATM SBus adapters. SunATM Interim API Support Software (SUNWatma) is only needed if you want to use the Application Programmers’ Interface (API).

4. Decide whether you want to be prompted for configuration information during the software installation, or if you want to configure the software by hand.

```
How would you like to configure the software?

    1) Pkgadd prompts user for information during install
         and automatically edits configuration files

    2) Files are manually edited by user after pkgadd ends
         (minimal user interaction is required during pkgadd)

Choose one [1-2,?,q]
```

- If you want to be prompted for configuration information, type 1, and continue with Step 5.
- If you want to manually edit the SunATM files, type 2, and complete the software installation. You must then edit the appropriate SunATM files on your system, as described in Chapter 4, and reboot your system as described in Section 3.3, “Rebooting the System and Examining Network Interfaces.”

Warning – You *must* edit these files before you reboot your system.

5. **When prompted, type in the version of the ATM Forum User Network Interface (UNI) Specification that will be used for each SunATM interface:**

Note – During the software installation, when you are are responding to the configuration questions, you can receive more information by answering with a question mark.

```
Two versions of the ATM Forum User Network Interface
Specification are supported in SunATM 2.0, UNI 3.0 and 3.1.

Which version will be used on ba0? [3.0] [?,q] 3.0
```

For each SunATM interface on your system, you will need to respond to a series of questions about your Internet protocol and hardware configuration. The following example will show the software installation of a single SunATM interface (`ba0`), which will be configured for both Classical IP and LAN Emulation.

6. **Select what type of Internet Protocol is required for your network:**

```
Configuring ba0:

What Protocol(s) will be used to support IP on ba0?

    1) Classical IP
    2) LAN Emulation
    3) Both Classical IP and LAN Emulation
    4) No IP Support

Choose one [1-4,?,q] 3
```

You will be prompted for different questions for both protocols. In this example, both types of protocols will be configured.

7. **Respond to the questions about the Classical IP network configuration.** To select the default value (displayed between brackets), press the Return key. Also, answer with a question mark to find out more about how to answer a specific question.

```
Will ba0 be used as an ARP Server? [no] [y,n,?,q]

What ATM address will be used on ba0 for Classical IP?

The switch prefix will be the first 13 bytes.

Enter the local portion of the ATM address. [local MAC address] [?,q]

What is the ATM address of the ATM ARP Server?

Will localswitch_server be used as the ATM ARP server address? [y] [y,n,?,q]

What IP hostname will be used on ba0 for Classical IP? mentos

IP address 128.146.90.133 will be used on ba0 for Classical IP.
```

Note – You cannot respond to these question using SunATM specific configuration variables. Because ATM addresses can be long, the SunATM software allows you to use variables when configuring the SunATM resource files by hand. See Section 4.4.2, “Using Variables in the /etc/aarconfig File,” on page 4-9, for more information.

8. **Respond to the questions about the LAN Emulation configuration.**

```
What ATM address will be used on ba0 for LAN Emulation?

The switch prefix will be the first 13 bytes.

Enter the end system identifier (esi). [local MAC address] [?,q]

What IP hostname will be used on ba0 for LAN Emulation? crumb

What IP address will be used on ba0 for LAN Emulation? [?,q] 128.146.90.134

What Emulated LAN Name will be used on ba0? [name supplied by LECS] [?,q]
```

Note – Repeat Steps 7 and 8 for each SunATM interface (sa and ba) on your system.

Note – The installation script will configure the SunATM based on the information you entered in Steps 5 through 8. If you need to update or reconfigure the SunATM resource files, see Chapter 4, “Configuring the SunATM Interface” for more information.

These packages will be installed as follows:

- SunATM Device Drivers (SUNWatm) will go into /kernel/drv
- SunATM Runtime Support Software (SUNWatmu) will go into /opt/SUNWatm/bin
- 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.

9. **Eject the SunATM 2.0 CD-ROM.**

- If your system is running the Volume Management software and the OpenWindows™ interface, click on the Eject Disk button on the `/cdrom/sunatm_2_0` 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.

10. **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 Troubleshooting During Software Installation

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.

During the software installation, when you are are responding to the configuration questions, you can receive more information by answering with a question mark.

For additional troubleshooting information, see Appendix C, “Troubleshooting and Error Messages.”

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
```

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), 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-155 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.

```
zardoz% /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 ffffff00 broadcast 129.144.130.255
        ether 8:0:20:75:89:ff
```

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

```
zardoz% 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-155 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 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 4

This chapter describes how to configure the SunATM 2.0 software to suit your network.

4.1 SunATM-155 SBus Adapter Versions

The SunATM 2.0 software supports earlier versions of the SunATM-155 SBus adapters, which were shipped with the SunATM 1.0 software.

If you are upgrading the SunATM 1.0 software on an older SunATM-155 adapter, you will still have an `sa` driver, and the interfaces will be referred to as `sa`*N*, where *N* is the instance number.

If you have new SunATM-155 SBus adapters 2.0, which are shipped with the SunATM 2.0 software, it will use the `ba` driver, and the interfaces will be referred to as `ba`*N*, where *N* is the instance number.

The correct software will be installed for your interfaces when you install using `pkgadd`, as described in Chapter 3. However, if you are doing any of the manual configuration described in the rest of this chapter, you must identify your interfaces properly. You can determine what type of board you have installed by examining the output of the `prtconf` command. It will indicate all instances of `ba` and `sa` devices that are installed in the system. In all the configuration examples that follow in this chapter, `ba` and `sa` should be used interchangeably, depending on the type of interfaces you have installed in your system.

Note – Although multiple SunATM-155 adapters may be installed in a system (on separate SBuses), no more than one SunATM 1.0 (`sa` driver) adapter in a single system can be connected to the same ATM switch. However, multiple SunATM 2.0 (`ba` driver) in the same system may be connected to the same switch.

4.2 Selecting SONET or SDH Framing Interface

The default framing interface is SONET, however, the SDH interface is also supported. Your switch product information should indicate whether your switch uses SONET or SDH. If the switch uses SDH, you will need to configure the SunATM interface to use SDH.

To change the framing interface from the default SONET to SDH:

1. **Add the following line to the** `/etc/system` **file:**

```
set ba:atm_sdh = 1
```

2. **Reboot the system.**
   Changes will not be in effect until the system is rebooted.

4.3 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 entry per interface. An entry contains the following fields:

Table 4-1 /etc/atmconfig Field Descriptions

Field	**Description**
Interface	The physical interface, ba*N* or sa*N*.
UNI	The version of the UNI specification used for signalling, 3.0 or 3.1.
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 lane*N* where *N* is the LAN Emulation instance number. Note that the LAN Emulation 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.

4.3.1 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 sa0 interface.
- An interface that supports both Classical IP and LAN Emulation on ba0, using UNI3.1. The LAN Emulation interface name is lane1.
- A Classical IP interface, supporting UNI 3.0, on ba1.

```
#Interface     UNI        CIP_Host     LANE_Instance     LANE_Host
#-------------------------------------------------------------------
 sa0           3.1        -                  0           atm0
 ba0           3.1        atm1               1           atm2
 ba1           3.0        atm3               -           -
```

4.4 Configuring a Classic Internet Protocol Interface

Classical IP, specified by RFC 1577, is one way of supporting the TCP/IP and UDP/IP protocols in an ATM environment. An ATM ARP server is used to resolve IP addresses to ATM addresses, replacing the traditional ARP protocol; 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 5.2, “Classical Internet Protocol.”

One major reason for the use of ATM ARP instead of the traditional ARP is because Classical IP does not support broadcast. Because of this, `ypbind` with the `-broadcast` option cannot be used to automatically locate the ypserver on a Classical IP ATM subnet.

If you are planning to run NIS over ATM, you must specify the list of ypservers via the `ypinit -c` command. See the `ypinit` man page for details of setting up the ypserver. Note that the IP addresses of the ypservers must be available in the `/etc/hosts` file.

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

4.4.1 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 `/opt/SUNWatm/bin` directory. The `aarsetup` program must also be run again if the physical configuration is changed, such as when a host is moved to a different switch.

Every node, or client, will have both an IP address and either an ATM address or a virtual circuit identifier (VCI). See Section 5.2.1, “ATM Address Resolution,” for ATM addressing scheme information.

Table 4-3 on page 4-8 lists the flags and the options they provide. All the flags required for most standard configurations are described in this section. If you have unusual configuration requirements, such as back-to-back connections over SVC, or interoperability issues, see Appendix F, “Advanced Configurations.”

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 4-2 `/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. *Hostname* should not appear; *ATM Address* should be provided if and only if SVCs are used. If an *s* entry is provided to use an ARP server (see below), *ATM Address* must be provided (a server is meaningful only in an SVC environment). See Table 4-3.
`L`	Represents the IP and ATM address of the local interface on an ARP server. *Hostname* should not appear; *ATM Address* is required. See Table 4-3.
`s`	Specifies a connection to the ATM ARP server. Either *ATM Address* or *VCI* (in the case of a PVC connection) should appear, but not both. *Hostname* should not appear. The *s* entry is required on all clients that need to communicate with the server for ATM address resolution. See Table 4-3

Table 4-2 `/etc/aarconfig` File Flags *(Continued)*

Flag	Description
`t`	Represents an IP to ATM address/VCI entry. `aarsetup` adds these entries into the local table. Any *t* entries on the server must contain *ATM Address* and may also contain *VCI* if PVC communication between the server and client is desired. In addition, there are some cases when a *t* 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 *t* entry containing *VCI*; or if a client wishes to cache frequently used addresses to avoid frequent ARP requests, a *t* entry containing *ATM Address* may be provided. See Table 4-3. **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 *t* entires, either by using the IP address in the *Hostname* field of the *t* entry, or by adding an entry to the local `/etc/hosts` file.
`a`	Represents an address that may have access to this host. If no *a* entries appear in the `aarconfig` file, access to the host is unrestricted. Including *a* 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 `$anymacsel`, 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. *Hostname* and *VCI* should not appear; *ATM Address* is required. See Table 4-3.

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 4-3 lists the flags and the options that they support.

Table 4-3 /`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

[1] **or – Means one or the other required, but using both is also legal.**
[2] **xor – 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.

Note – Additional flags that can be used for advanced configurations, such as back-to-back connections over SVC, or to change the behavior of the interface, are described in Appendix F, "Advanced Configurations."

4.4.2 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 4-8.

Table 4-4 Predefined SunATM Variables

Variable	Description
`prefix`	The 13-byte prefix associated with the local switch.
`mac`	The 6-byte medium access control (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:$anymacsel`. However, it is *not* the same as `$prefix:$anymacsel:0?`, which requires that the first digit of the selector byte is a `0`.
`sunmacsel`*`N`*	The concatenation of one of a series of reserved MAC addresses and `$sel` to create a block of reserved ATM ARP server addresses. *N* should be a decimal number in the range 0 - 199.
`localswitch_server`	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.

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 `sunmacsel`*N* variables can be used in conjunction with a prefix as well known server addresses which are not bound to a particular system. As an example, consider the case where a server which 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 instead 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 `sunmacsel`*N* 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`, `sunmacsel`*N* (where *N* is a number between 0 and 199), and `localswitch_server`.

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

`localswitch_server`	`08:00:20:75:48:08`
`sunmacsel`*N* `base`	`08:00:20:75:48:10`

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

4.4.3 *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          -       $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   45:00:00:00:00:00:00:00:0f: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       -       45:00:00:00:00:00:00:00:0f:00:00:00:00:$anymacsel          -      a
```

4.5 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 5.3, "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 4.4, "Configuring a Classic Internet Protocol Interface," do not affect LAN Emulation interfaces.

4.5.1 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 `/opt/SUNWatm/bin` directory. The `lanesetup` program must also be run again if the physical configuration is changed, such as when a host is moved to a different switch.

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   ATM Address      VCI       Flag
```

These entry fields are described in Table 4-5.

Table 4-5 `/etc/laneconfig` Entry Descriptions

Field	**Description**
`Interface`	Refers to the LAN Emulation interface, lane*N*.
`MAC Address`	This field is for the 6-byte MAC address of the interface.
`ATM Address`	This field is for the 20-byte ATM address. The `$myaddress` 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 (`l`), a permanent table entry (`t`), or an LECS address entry (`c`). For a complete description of the `laneconfig` flags, see Table 4-6.

Table 4-6 `/etc/laneconfig` Flag Descriptions

Flag	**Description**
`l`	This flag designates a local address entry. There must be an *l* 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 *t* entry for that system; you may also create PVCs with a `t` 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.

Table 4-7 describes the required, optional, and illegal fields for each flag type.

Table 4-7 `laneconfig` Flag Requirements and Options

Interface	MAC Address	ATM Address	VCI	Flag
required	illegal	required	illegal	`l`
required	required	xor[1]	xor[1]	`t`
required	Emulated LAN name	illegal	illegal	`n`

[1] xor 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.

4.5.2 *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 4-8. For a complete description of how to use these variables, see Section 4.4.2, "Using Variables in the /etc/aarconfig File."

Table 4-8 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.

4.5.3 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      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      ATM_Address         VCI      Flag
lane0         -                $myaddress          -           l
lane0         elan1            -                   -           n
```

Classical IP and LAN Emulation Protocols 5

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

5.1 ATM Addresses and Address Registration

UNI signaling uses ATM addresses for signaling. 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.

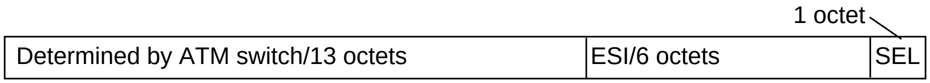


Figure 5-1 ATM Address Fields

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`.

5.1.1 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.

5.2 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.

5.2.1 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 signaling 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 4-3 on page 4-8 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 4.4.1, "Editing the /etc/aarconfig File," on page 4-5.

The ATM Address Resolution software contains several timers which control its resource usage. Two key timers include the timer controlling the teardown of inactive connections to peers or the server, and the timer controlling the deletion from the local kernel table of ARP entries received from peers or the server. You may choose to lengthen or shorten these timers based on a particular network's needs. This is done with system variables, which may be added to the `/etc/system` file. After changing this file, you must reboot the system for the changes to take effect. Set the variables as follows:

```
set aar:aar_max_quiescent_qcc = m
set aar:aar_max_quiescent_ace = n
```

where *m* and *n* represent the number of 30 second intervals each timer should last. The first variable, `aar_max_quiescent_qcc`, determines how many intervals to wait before tearing down a connection to a peer or server. The receipt or transmission of a packet on the connection resets the timer. The second variable, `aar_max_quiescent_ace`, determines how many intervals to wait before removing an `arp` cache entry from the local table. This applies only to entries that are created via the ATM ARP process, and no to those entries that are loaded from the `/etc/aarconfig` file. Receipt or transmission of a packet from or to the peer represented by the entry resets the timer.

5.2.2 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 cirtuit (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 4.4.2, "Using Variables in the /etc/aarconfig File," on page 4-9, 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`.

5.3 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 2.0 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.

5.3.1 LAN Emulation Services

5.3.1.1 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.

5.3.1.2 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.

5.3.1.3 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.

5.3.2 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.

5.3.3 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 (the command `qccstat(1)` may be used to view all existing connections for a given interface).

Connection To	**Originator**	**Comments**
LECS	host	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.
LES	host	Point-to-point connection over which the host may send LE ARP requests and receive responses from the LES.
LES	LES	Point-to-multipoint connection over which the LES may send administrative information to all hosts. Hosts may not send on this connection.
BUS	host	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	BUS	Point-to-multipoint connection over which the BUS sends broadcast messages. Hosts may not send on this connection.

Wiring Scheme and Pin Descriptions

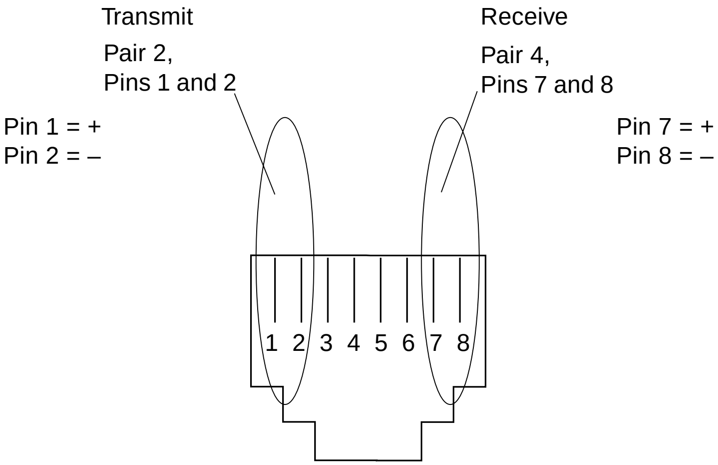


Figure A-1 Designation T568B

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-155 SBus Cards Specifications

B

B.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.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.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.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	0 to 95% non-condensing
	(40°C, wet bulb temperature)	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)	1.0g (5 to 500 Hz)
	(Sweep Rate = 1 octave/min.)	(Sweep Rate = 1 octave/min.)

Troubleshooting and Error Messages C

C.1 Troubleshooting

C.1.1 Software Installation

The software installation scripts have been designed to make configuration of your SunATM adapter as easy as possible. However, there are many new concepts involved in an ATM network, and some of the required information may be unfamiliar. Also, your configuration may have special requirements that are not covered by the installation scripts. The following tips and pointers should help you get through the installation process more easily.

- If you are unsure of response formats *or* if you get an "Invalid Response" message in response to your input:

  Try typing **?** at the input prompt. This will give you a brief description of the information required and the default values, if applicable, as well as an example response.

- If you are not familiar with the type of information you are being asked during the software installation:

  Check the Glossary at the end of this manual. It contains descriptions of all of the terms used in the installation script.

- If the installation script exits because it finds no SunATM SBus adapter installed:

  Make sure that your SunATM adapter is installed in the system, and that you performed a reconfiguration boot after installing the hardware. If you are not sure if you have done a reconfiguration boot since installing the hardware, perform the following steps as super-user to reboot with the reconfiguration option:

```
# cd /
# touch reconfigure
# sync
# init 6
```

- If you have several systems to install, and you do not want to re-type the same information for each system installation:

  If the configuration of the systems is similar, you can install once and have the scripts create appropriate files, then install without the autoconfiguration option on the other systems, and copy the automatically generated configuration files from the first system to all of the systems. This may or may not be an improvement, depending on how similar the configuration of each system is. Note that using system variables such as `$myaddress` make configuration files very portable. The only required configuration file is `/etc/atmconfig`. If you are running Classical IP or LAN Emulation, you must also have the `/etc/aarconfig` and/or `/etc/laneconfig` configuration files, respectively.

- If you accidently enter an incorrect value or name:

  You have two options in this case. You can either exit the script by typing **q** at any input prompt and then restart the installation, or continue with the installation and edit the appropriate file after the installation completes. Refer to Chapter 4, "Configuring the SunATM Interface," for a description of the configuration files and their contents. The Glossary in this manual also contains brief descriptions of the function of each file to help determine where the changes should be made.

C.1.2 Starting a SunATM Interface

There are many steps involved in making an interface active on an ATM network. Once your interface is configured properly, these steps should all happen transparently 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.1.2.1, "Generic Configuration"	Refers to all ATM configurations, regardless of the type of IP support involved, if any.
Section C.1.2.2, "Classical IP Configuration"	Refers only to interfaces configured to support Classical IP.
Section C.1.2.3, "LAN Emulation Configuration"	Refers to interfaces configured to support LAN Emulation.

C.1.2.1 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 4.3, "Editing the /etc/atmconfig File," on page 4-3, 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 signaling 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 signaling.
   The SunATM 2.0 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 5.1, "ATM Addresses and Address Registration," on page 5-2, 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.

3. If you have two `sa` devices (the SunATM 1.0 SBus adapters), they must not be connected to the same switch.
   Like many ethernet interfaces, the SunATM-155 1.0 adapters (`sa`) do not have unique MAC addresses; they use the MAC address assigned to the system in which they are installed. The MAC address is used as the local "ESI" portion of the ATM address registered for each interface; therefore, if two `sa` devices are in the same system, they will have the same MAC address, and will each attempt to register the same address. If they are connected to the same switch, this will result in the successful registration for the first interface to try, and failure for the second. The SunATM SBus adapters 2.0 have their own MAC addresses, and thus do not have this limitation.

- 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.1.2.2 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, either `sa`*N* or `ba`*N*, 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 4.3, "Editing the /etc/atmconfig File," and Section 4.4.1, "Editing the /etc/aarconfig 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.1.2.1, "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 signaling, 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 signaling (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.1.2.3 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, `lane`*N*, 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 4.3, “Editing the /etc/atmconfig File” and Section 4.5.1, “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 4.5.1, "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/laneconfig`. See Section 4.5.1, "Editing the /etc/laneconfig 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 kB 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 signaling (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 S60sunatm

```
Cannot find ATM utilities in /opt/SUNWatm/bin
```

The SunATM utility directory /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 3-11, for more information). If necessary, the package may need to be re-installed.

```
Cannot load ATM Drivers
```

```
Please remove /etc/hostname.sa* and/or /etc/hostname.ba*
and reboot your machine
```

This message is produced by the setup script, S60sunatm. It is printed if it finds a file that matches hostname.sa* or hostname.ba* in /etc; the problem can be solved by removing any such files and rebooting the system.

`Please install <SUNWatm>` or

```
Please install <SUNWatmu>
```

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

```
warning: extra entry for <device> being ignored
```

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

```
warning: ignoring extra entry for <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 4.3, “Editing the /etc/atmconfig File”), filling in all the appropriate information, and reboot the system. Empty fields should be indicated with a hyphen (`-`).

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 4.4.1, “Editing the /etc/aarconfig File,” on page 4-5, 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 4.4.2, “Using Variables in the /etc/aarconfig File,” on page 4-9, and Section 4.5.2, “Using Variables in the /etc/laneconfig File,” on page 4-17, 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/laneconfig`, 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/laneconfig`, 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.

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 D

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 configuration files for the SunATM SNMP Management Console and the SunATM SNMP agent daemon is contained in the `SUNWatmu` package. The management console configuration files will be installed in `/opt/SUNWatm/snmp`, and the agent daemon will be installed in `/opt/SUNWatm/bin`. Required configuration files for the agent daemon will be installed in `/etc/atmsnmp`.

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 to run as an SNMP agent, you must choose the SNMP agent option when installing the `SUNWatmu` package on your system. If this option is selected, the software will be configured to start the SunATM SNMP daemon (`atmsnmpd`) during system boot.

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/armsnmp/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.

Note – Only one SNMP agent may be active on a system. If you wish to run the SunATM agent on a host, no other SNMP agents may run on that host. If other agents are running, the `atmsnmpd` daemon will exit with an error message.

Application Programmers' Interface E

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 interim API that Sun provides:

- Will cover Q.93B to set up and tear down connections
- Will also deal directly with the driver

Each API set, Q.93B and driver, contains both a user API and a kernel API.

The 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 the messages required to set up and tear down connections. Table E-2 lists those message types. Figure E-2 shows the message format that should be used by kernel applications; user space applications should use two `strbuf` structures instead of the two `mblks`.

The qcc man pages that provide more details on how to use these functions are:

user space	qcc_bld (3), qcc_parse (3), and qcc_len (3)
kernel space	qcc_bld (9F) and qcc_parse (9F)

The man page for the signaling driver is q93b (7).

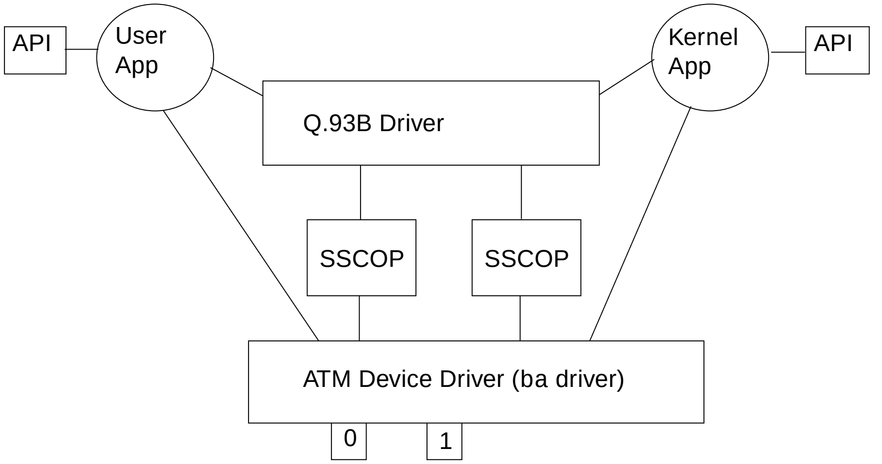


Figure E-1 ATM Signaling

E.1 Q.93B API

The Q.93B driver is an M-to-N MT safe (D_MP) mux driver. On the upper side, the Streams interface is the Q_Primitives that consists of two mblocks. The first mblock is the header of the type M_PROTO, and the second mblock is the raw Q.93B message of the type M_DATA.

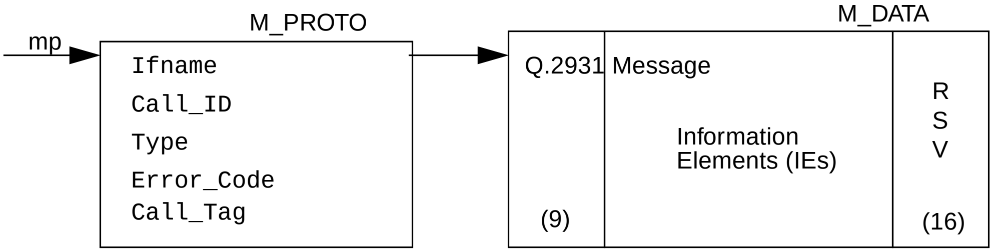


Figure E-2 Message Format

Table E-1 Message Meanings

Message	**Explanation**
`Ifname`	A null-terminated string containing the device name (for example, sa0)
`Call_ID`	A unique number from Q.93B per interface.
`Type`	The same as the Q.93B message type except there is a local Non-Q.93B message type SETUP_ACK. The SETUP_ACK message is used to provide the Call_ID to the user.
`Error_Code`	The error returned from Q.93B when an erroneous message is received from the user. The exact same mblock chain shall be returned to the user with the Error_Code field set. The user must always clear this field
`Call_Tag`	A number assigned by the calling application layer to a SETUP message. When a SETUP_ACK is received from Q.93B, the Call_ID has been set; the Call_Tag field may be used to identify the ack with the original request. From that point on, the Call_ID value should be used to identify the call.

The upper layer shall leave the Q.93B header portion (9 bytes) of the Q.93B message in the second mblock blank. The Q.93B driver will fill in the Q.93B header. The upper layer should reserve 16 bytes at the end of the second mblock for the layer 2 protocol for performance.

E.1.1 Q.93B Driver

Table E-2 Messages Between the User and the Q.93B 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 Q.93B to user;
DOWN is from user to Q.93B

Setup

When the user decides to make a call, the user sends a SETUP message down to Q.93B and waits for a SETUP_ACK from Q.93B. After SETUP_ACK is received, the user waits for either a CALL_PROCEEDING, CONNECT, or RELEASE_COMPLETE message from Q.93B (Q.93B ignores all other messages). After the CONNECT message is received, the user can use the virtual channel.

When the user receives a SETUP message from Q.93B, the user shall respond with either a CALL_PROCEEDING, CONNECT, or RELEASE_COMPLETE message to Q.93B. After the CONNECT_ACK message is received, the user can use the virtual channel.

Release Procedure

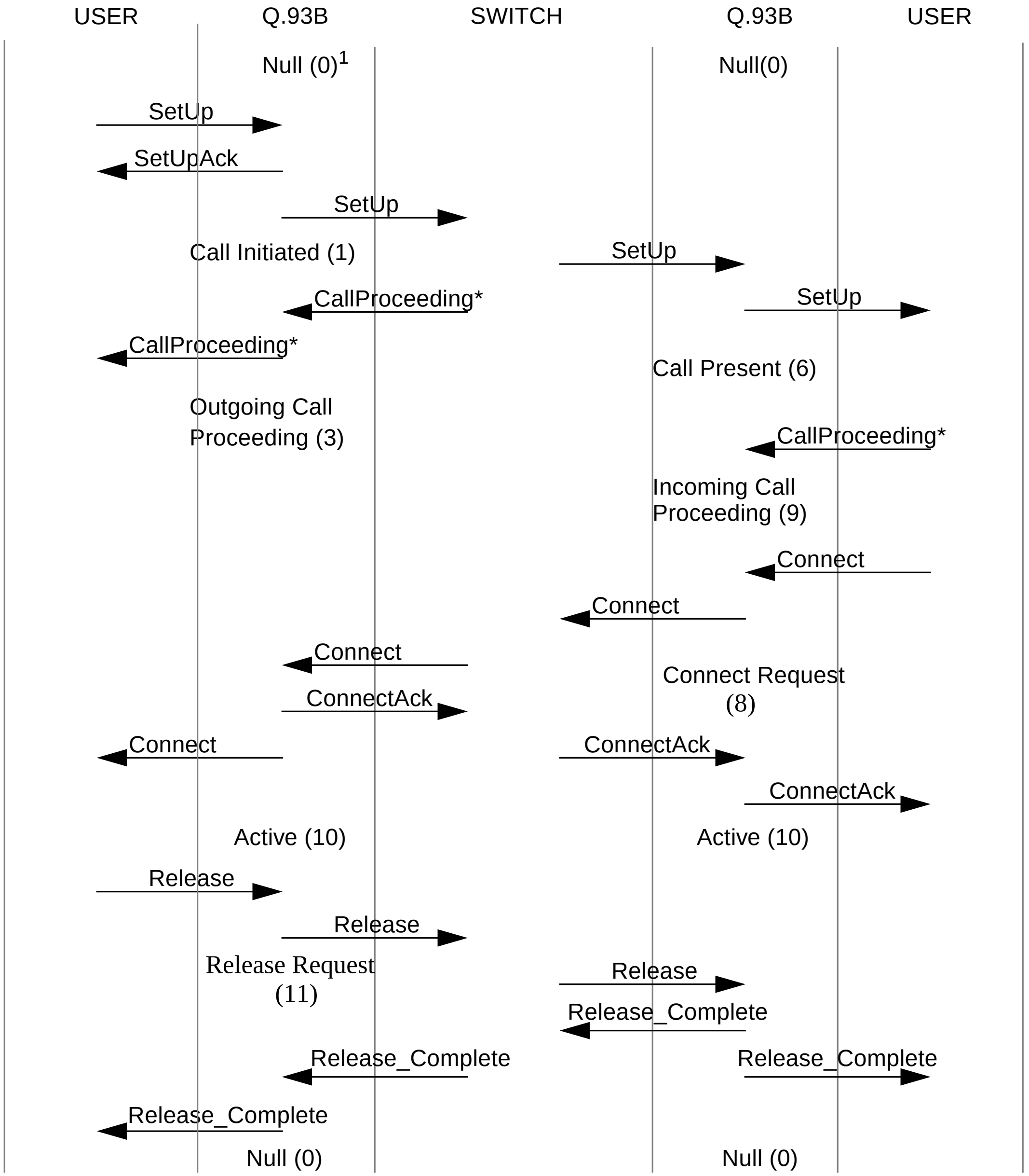
To clear an active call or a call in progress, the user sends a RELEASE message down to Q.93B and waits for a RELEASE_COMPLETE from Q.93B. Anytime the user receives a RELEASE_COMPLETE message from Q.93B, the user shall release the virtual channel if the call is active or in progress.

Q.93B never sends a RELEASE message to the user. The user only sends the RELEASE_COMPLETE message right after sending a SETUP_ACK message to Q.93B to reject the call in response to a SETUP message from Q.93B. At any other time, to reject or tear down a call, the user shall send a RELEASE message to Q.93B.

Exception Conditions

When a message is received from the user with an unattached port, Q.93B shall return the message with the Error_Code BAD_PORT.

Normal Call Set up and Tear Down



[1] XX(n): Q.93B State Name (Q.93B State Number)

* Optional

E.1.2 Q.93B User Space API

Q.93B user space consists of all `qcc_bld`, `qcc_len`, `qcc_parse`, and `qcc_util` functions under section 3.

See the `qcc_bld(3)`, `qcc_len(3)`, `qcc_parse(3)`, and `qcc_util(3)` man pages.

E.1.3 Q.93B Kernel Space API

Q.93B kernel space consists of all `qcc_bld` and `qcc_parse` functions under section 9F.

See the `qcc_bld(9F)` and `qcc_parse(9F)`man pages.

E.1.4 Driver API

See the `ba` man page.

Note – Man pages are contained in the SUNWatmu package, and they will go into `/opt/SUNWatm/man` directory (add this path to your system's `$MANPATH` environment variable). Interim API examples will go into `/opt/SUNWatm/examples`.

Advanced Configurations F

Networks are rarely homogeneous. For interoperability purposes, there may be cases when a network must be configured with:

- Different characteristics than the defaults that are built into the SunATM-155 SBus adapter
- Unusual addressing schemes that require more than the basic flags described in Section 4.4.1, “Editing the /etc/aarconfig File.”

Edit the `/etc/aarconfig` file or the `/etc/laneconfig` file using the flags in this section to alter the defaults and/or change the behavior of the interface.

F.1 Flags That Specify Additional Entry Types in the `/etc/aarconfig` File

`b` Specifies the VCI to use for back-to-back SVC connections between two ARP clients. This entry is required in addition to the *t* entry because the VCI is normally provided by the switch. *VCI* is required. See Table F-1.

`B` Specifies the VCI to use for a back-to-back SVC connection between an ARP client and a server. This entry is required in addition to the *s* or *t* entry. *VCI* is required. See Table F-1.

Note – The *b* and *B* options are useful for testing. For normal operation, the *t* flag may be used with a VCI and no ATM address.

Note – When using SVCs over back-to-back connections, the two systems should use different VCI values. Also, if both a *b* and a *B* entries are used, they should each have different VCI values.

`c` Indicates an alternate client address for ARP traffic only in the server's `aarconfig` file. There may be configurations that require an ATM ARP client to have different ATM addresses or PVCs for ARP connections and for regular data connections. In this case, a distinction must be made in the server's `/etc/aarconfig` file between the two address entries; the *c* flag specifies the ARP address, while the *t* flag identifies the data address. If both a *t* and *c* entry are provided in a server's `aarconfig` file, any of the VCIs or ATM addresses in those entries may be used by the client to contact the server. Either *ATM Address* or *VCI* is required. See Table F-1.

As an example, consider this situation: a server has a client that uses a different selector byte to identify the ARP connection. A requirement is that the ATM address end with 00 for data connections, and end with 05 for ARP connections. To represent this client, the following two entries are required in the server `/etc/aarconfig` file:

```
Interface  Host     ATM Address                                              VCI Flag
sa0       client1   45:00:00:00:00:00:00:00:0f:00:00:00:00::08:00:20:13:00:10:00  -    t
sa0       client1   45:00:00:00:00:00:00:00:0f:00:00:00:00::08:00:20:13:00:10:05  -    c
```

A Specifies on the server an alternate local ATM address for ARP traffic only. Similar to situations where a client has different addresses for data and ARP connections (see above example), it may be that the server also has different addresses. In this case, it is still sufficient to have only an *s* entry in the client `/etc/aarconfig` file since the client will be able to send ARP requests to the server for the server's data address. However, the server must be aware of the two different addresses. This is accomplished by using the *A* flag to identify the ARP address, while the *L* flag identifies the data address. *ATM Address* is required. See Table F-1.

As an example, consider a situation where the server uses an address with selector 00 for data connections and an address with selector 01 for ARP connections:

```
Interface  Host     ATM Address                                           VCI  Flag
sa0          –     45:00:00:00:00:00:00:00:0f:00:00:00:00::08:00:20:13:00:10:00   -    L
sa0          –     45:00:00:00:00:00:00:00:0f:00:00:00:00::08:00:20:13:00:10:01  64    A
```

Table F-1 /`etc/aarconfig` Advanced Configuration Flags

Interface	**Host**	**ATM Address**	**VCI**	**Flags**
required	illegal	illegal	required	b
required	illegal	illegal	required	B
required	optional	or[1]	or[1]	c
required	illegal	required	illegal	A
required	illegal	illegal	illegal	P
required	illegal	illegal	illegal	I
required	no_ilmi	illegal	illegal	o

[1] or – Means one or the other required, and both are also legal.

F.2 Flags That Specify Additional Entry Types in the /etc/laneconfig File

c This flag allows an alternate LECS address to be specified. By default, the SunATM softwares 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.

Table F-2 /etc/laneconfig Advanced Configuration Flags

Interface	**Mac_Address**	**ATM Address**	**VCI**	**Flags**
required	illegal	xor[1]	xor	c
required	illegal	xor[1]	xor	s
required	illegal	illegal	illegal	P
required	no_ilmi	illegal	illegal	o

[1] xor means that you an used either the ATM Address field or the VCI field, but not both.

F.3 Flags That Change the Behavior of the Interface

P Enables the function that sends a Call_Proceeding message when setting up a connection (this message type is optional according to the UNI 3.0 Specification). Some switches may not be designed to handle this message type since it is not required, so default behavior of the SunATM signaling is to *not send* the message.

Use of the Call_Proceeding message is desirable if some amount of delay is likely in the processing of setup messages because it essentially prolongs the length of time a caller will wait to receive a connect back before giving up on the connection.

If your switch supports the Call_Proceeding message, and the feature is desirable, you can turn the function on for a particular interface using the *P* flag.

I Provides additional security for Classical IP. If your ARP server is capable of handling inverse ARP requests (the SunATM implementation has this capability), you may choose to have a client who receives a setup request from a peer do address verification with the server rather than the calling party. The default behavior is to send the inverse ARP for address verification to the calling party. If the *I* flag is set in the `/etc/aarconfig` file for a particular interface, the inverse ARP will be sent to the server instead. This allows the access list for the network to be specified in the server `/etc/aarconfig` file. Hosts that do not appear in this file will not be verified by the inverse ARP sent to the server, so the call will not be accepted.

The default behavior for all interfaces, rather than just one, may be set by adding one or both of the following lines to the `/etc/system` file:

```
set aar:aar_can_use_call_proc = 1

set aar:aar_invarp_to_server = 1
```

These are equivalent to providing a *P* or *I* flag entry for every interface in `/etc/aarconfig`.

o Notifies the system that ILMI should not be supported on this interface. If your interface is connected to a switch that does not support ILMI, add this option to your `/etc/aarconfig` or `/etc/laneconfig` file. Note that the variables `prefix` and `localswitch_server` (which use the switch prefix obtained from the switch via ILMI) may not be used if ILMI is not running.

Running Diagnostic Tests G

The diagnostic tests available for the SunATM-155 adapters is the `nettest`, which is included in both the SunDiag™ system exerciser and the SunVTS™ Validation and Test Suite.

G.1 SunDiag System Excerciser and SunVTS Validation and Test Suite

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

If you are running the Solaris 2.4 operating environment, you can find the `nettest` on the SunDiag system exerciser. In the SunDiag environment, you must run `nettest` from the command line.

If you are running the Solaris operating environment, revision 2.5 or higher, you can find the `nettest` on the SunVTS Validation and Test Suite. Like the SunDiag software, the SunVTS software runs diagnostic tests on Sun Systems. The SunVTS OpenWindows interface supports `nettest` on ATM devices.

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

G.2 Using the OpenBoot PROM Selftest

The SunATM-155 SBus adapters 2.0 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-155 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-155 Adapter's Selftest

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

```
SPARCstation 10, Type 4 keyboard
Rom Rev 2.4, 16MB memory installed
Ethernet address 8:0:20:8;42:7, Host ID 51000007

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-155 SBus adapter, look for the `ba` device in the list of devices . In the following example, the `ba` device is in **bold** type.

```
ok show-devs
/TI,TMS390Z50@f,f8fffffc
/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/eeprom@0,200000
/obio/zs@0,0
/obio/zs@0,100000
/iommu@f,e0000000/sbus@f,e0001000
/iommu@f,e0000000/sbus@f,e0001000/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
/packages/obp-tftp
/packages/deblocker
/packages/disk-label
<#0> ok
```

3. **To run the SunATM-155 SBus adapter's selftest, type `test` and the pathname to the** `ba` **device:**

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

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` configuration file.

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 and 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 signaling.

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 signaling 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 Signaling 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 signaling 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*.

sunmacsel*N*

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; signaling uses VC 5; *ILMI* uses VC 16, and *LAN Emulation* connections to the *LECS* use VC 17.

Index

Reader Comments

We welcome your comments and suggestions to help improve this manual. Please let us know what you think about the *SunATM SBus Adapters Manual,*
part number *802-4439-10*.

- The procedures were well documented.

Strongly Agree	Agree	Disagree	Strongly Disagree	Not Applicable
❑	❑	❑	❑	❑

Comments ____

- The tasks were easy to follow.

Strongly Agree	Agree	Disagree	Strongly Disagree	Not Applicable
❑	❑	❑	❑	❑

Comments ____

- The illustrations were clear.

Strongly Agree	Agree	Disagree	Strongly Disagree	Not Applicable
❑	❑	❑	❑	❑

Comments ____

- The information was complete and easy to find.

Strongly Agree	Agree	Disagree	Strongly Disagree	Not Applicable
❑	❑	❑	❑	❑

Comments ____

- Do you have additional comments about the *SunATM SBus Adapters Manual*?

____

____

____

____

Name: ____

Title: ____

Company: ____

Address: ____

____

Telephone: ____

Email address: ____

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 1 MOUNTAIN VIEW, CA

POSTAGE WILL BE PAID BY ADDRESSEE

SUN MICROSYSTEMS, INC.
Attn: Manager, Hardware Publications
MS MPK 14-101
2550 Garcia Avenue
Mt. View, CA 94043-9850