

# LAN Configuration and Management Manual

## Abstract

This manual describes how to configure, operate, and manage the ServerNet LAN Systems Access (SLSA) subsystem on an HP Integrity Nonstop™ NS-series or HP NonStop S-series server. This manual includes detailed descriptions of the Subsystem Control Facility (SCF) commands used with the SLSA subsystem and a quick-reference section showing SCF command syntax.

## Product Version

SLSA G06

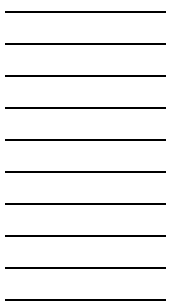
## Supported Release Version Updates (RVUs)

This manual supports G06.24 and all subsequent G-series RVUs and H06.03 and all subsequent H-series RVUs and J06.03 and all subsequent J-series RVUs until otherwise indicated by its replacement edition.

Part Number	Published
520469-011	August 2010

**Document History**

<b>Part Number</b>	<b>Product Version</b>	<b>Published</b>
520469-007	SLSA G06	July 2005
520469-008	SLSA G06	September 2005
520469-009	SLSA G06	May 2006
520469-010	SLSA G06	November 2006
520469-011	SLSA G06	August 2010



# LAN Configuration and Management Manual

<a href="#">Glossary</a>	<a href="#">Index</a>	<a href="#">Examples</a>	<a href="#">Figures</a>	<a href="#">Tables</a>
--------------------------	-----------------------	--------------------------	-------------------------	------------------------

- [What's New in This Manual](#)   vii
  - [Manual Information](#)   vii
  - [New and Changed Information](#)   viii
  - [HP Encourages Your Comments](#)   ix
- [About This Manual](#)   xi
  - [Who Should Use This Manual](#)   xi
  - [What's in This Manual](#)   xi
  - [Where to Get More Information](#)   xii
  - [Notation Conventions](#)   xii
  - [Abbreviations](#)   xvii

## 1. Configuration Quick Start

- [Initial SLSA Subsystem Configuration](#)   1-1
  - [Configuration File Contents](#)   1-2
- [Verifying the Initial SLSA Subsystem Configuration](#)   1-2
  - [Checking the State of the LAN Manager \(LANMAN\) Process](#)   1-3
  - [Checking the State of the LAN Monitor \(LANMON\) Processes](#)   1-3
  - [Checking the State of the TCP/IP Processes](#)   1-3
  - [Checking the State of the SLSA Subsystem Objects](#)   1-4
- [Troubleshooting Tips](#)   1-5
  - [Starting the LANMAN Process](#)   1-5
  - [Starting a LANMON Process](#)   1-5
  - [Starting a TCP/IP Process](#)   1-6
  - [Starting the LAN Subsystem Objects](#)   1-6
- [Installing an Adapter](#)   1-7
- [Changing the Initial SLSA Subsystem Configuration](#)   1-8
  - [WAN Wizard Pro](#)   1-8

## **2. Introduction to the SLSA Subsystem**

<a href="#">Overview of SLSA</a>	2-1
<a href="#">ServerNet Architecture and SLSA</a>	2-5
<a href="#">The SLSA Role Within the System</a>	2-7
<a href="#">System-Level Processes and Libraries</a>	2-8
<a href="#">LAN Service Providers</a>	2-9
<a href="#">LAN Clients</a>	2-9
<a href="#">Architecture of the SLSA Subsystem</a>	2-10
<a href="#">SLSA Subsystem Management</a>	2-10
<a href="#">SLSA Subsystem SCF Objects</a>	2-11
<a href="#">Fault Tolerance of the SLSA Subsystem</a>	2-16
<a href="#">Loss of a ServerNet Fabric</a>	2-16
<a href="#">Loss of Access of One Processor to a SAC</a>	2-16
<a href="#">Loss of a Processor</a>	2-17
<a href="#">Loss of Access of All Processors to a SAC</a>	2-17
<a href="#">Loss of an Adapter</a>	2-18
<a href="#">Loss of Media</a>	2-18

## **3. SLSA Subsystem Installation and Configuration**

<a href="#">SLSA Subsystem Installation</a>	3-1
<a href="#">SLSA Subsystem Configuration</a>	3-2
<a href="#">HP Manufacturing Naming Conventions (G06.26 and Earlier RVUs)</a>	3-2
<a href="#">G4SA Naming Conventions (G06.24, G06.25, and G06.26 RVUs)</a>	3-4
<a href="#">HP Manufacturing Naming Conventions (For G06.27 and Later G-Series RVUs and HO6.03 and Later H-Series RVUs and J06.03 and Later J-Series RVUs)</a>	3-6
<a href="#">Suggested Naming Conventions for Adapters</a>	3-7
<a href="#">LAN Configuration</a>	3-9

## **4. SCF Commands for the SLSA Subsystem**

<a href="#">Supported Commands and Object Types</a>	4-2
<a href="#">Command Cancellation</a>	4-3
<a href="#">Sensitive and Nonsensitive Commands</a>	4-3
<a href="#">SCF Objects</a>	4-4
<a href="#">SCF Object States</a>	4-6
<a href="#">null Object Type</a>	4-6
<a href="#">Generic Processes</a>	4-6
<a href="#">ABORT Command</a>	4-7
<a href="#">ABORT ADAPTER Command</a>	4-7

## **4. SCF Commands for the SLSA Subsystem (continued)**

<a href="#"><u>ABORT ATMSAP Command</u></a>	4-9
<a href="#"><u>ABORT LIF Command</u></a>	4-10
<a href="#"><u>ABORT MON Command</u></a>	4-11
<a href="#"><u>ABORT PIF Command</u></a>	4-12
<a href="#"><u>ABORT SAC Command</u></a>	4-13
<a href="#"><u>ADD Command</u></a>	4-15
<a href="#"><u>ADD ADAPTER Command</u></a>	4-15
<a href="#"><u>ADD ATMSAP Command</u></a>	4-21
<a href="#"><u>ADD LIF Command</u></a>	4-23
<a href="#"><u>ALTER Command</u></a>	4-27
<a href="#"><u>ALTER ATMSAP Command</u></a>	4-27
<a href="#"><u>ALTER LIF Command</u></a>	4-29
<a href="#"><u>ALTER PIF Command</u></a>	4-32
<a href="#"><u>ALTER SAC Command</u></a>	4-36
<a href="#"><u>DELETE Command</u></a>	4-39
<a href="#"><u>DELETE ADAPTER Command</u></a>	4-39
<a href="#"><u>DELETE ATMSAP Command</u></a>	4-39
<a href="#"><u>DELETE LIF Command</u></a>	4-40
<a href="#"><u>INFO Command</u></a>	4-41
<a href="#"><u>INFO ADAPTER Command</u></a>	4-41
<a href="#"><u>INFO ATMSAP Command</u></a>	4-44
<a href="#"><u>INFO LIF Command</u></a>	4-46
<a href="#"><u>INFO MON Command</u></a>	4-49
<a href="#"><u>INFO PIF Command</u></a>	4-50
<a href="#"><u>INFO SAC Command</u></a>	4-60
<a href="#"><u>LISTOPENS Command</u></a>	4-63
<a href="#"><u>LISTOPENS Command</u></a>	4-63
<a href="#"><u>NAMES Command</u></a>	4-64
<a href="#"><u>NAMES null Command</u></a>	4-64
<a href="#"><u>NAMES ADAPTER Command</u></a>	4-67
<a href="#"><u>NAMES ATMSAP Command</u></a>	4-67
<a href="#"><u>NAMES LIF Command</u></a>	4-69
<a href="#"><u>NAMES MON Command</u></a>	4-71
<a href="#"><u>NAMES PIF Command</u></a>	4-71
<a href="#"><u>NAMES PROCESS Command</u></a>	4-72
<a href="#"><u>NAMES SAC Command</u></a>	4-75
<a href="#"><u>RESET Command</u></a>	4-75
<a href="#"><u>RESET ADAPTER Command</u></a>	4-76

## **4. SCF Commands for the SLSA Subsystem (continued)**

<a href="#"><u>START Command</u></a>	4-76
<a href="#"><u>START ADAPTER Command</u></a>	4-76
<a href="#"><u>START ATMSAP Command</u></a>	4-77
<a href="#"><u>START LIF Command</u></a>	4-79
<a href="#"><u>START MON Command</u></a>	4-79
<a href="#"><u>START PIF Command</u></a>	4-80
<a href="#"><u>START SAC Command</u></a>	4-81
<a href="#"><u>STATISTICS Command</u></a>	4-83
<a href="#"><u>STATS ATMSAP Command</u></a>	4-83
<a href="#"><u>STATS PIF Command</u></a>	4-84
<a href="#"><u>STATUS Command</u></a>	4-105
<a href="#"><u>STATUS ADAPTER Command</u></a>	4-106
<a href="#"><u>STATUS ATMSAP Command</u></a>	4-107
<a href="#"><u>STATUS LIF Command</u></a>	4-109
<a href="#"><u>STATUS MON Command</u></a>	4-111
<a href="#"><u>Considerations</u></a>	4-113
<a href="#"><u>STATUS PIF Command</u></a>	4-113
<a href="#"><u>STATUS PROCESS Command</u></a>	4-123
<a href="#"><u>STATUS SAC Command</u></a>	4-125
<a href="#"><u>STOP Command</u></a>	4-128
<a href="#"><u>STOP ADAPTER Command</u></a>	4-128
<a href="#"><u>STOP ATMSAP Command</u></a>	4-129
<a href="#"><u>STOP LIF Command</u></a>	4-130
<a href="#"><u>STOP PIF Command</u></a>	4-131
<a href="#"><u>STOP SAC Command</u></a>	4-133
<a href="#"><u>SWITCH Command</u></a>	4-134
<a href="#"><u>SWITCH PROCESS Command</u></a>	4-134
<a href="#"><u>TRACE Command</u></a>	4-135
<a href="#"><u>TRACE MON Command</u></a>	4-135
<a href="#"><u>TRACE PIF Command</u></a>	4-138
<a href="#"><u>TRACE SAC Command</u></a>	4-145
<a href="#"><u>VERSION Command</u></a>	4-149
<a href="#"><u>VERSION MON Command</u></a>	4-149
<a href="#"><u>VERSION null Command</u></a>	4-150
<a href="#"><u>VERSION PROCESS Command</u></a>	4-151

## **5. Managing the SLSA Subsystem**

<a href="#">Adding an Adapter to the System</a>	5-1
<a href="#">Stopping and Starting an Adapter</a>	5-4
<a href="#">Renaming an Adapter</a>	5-7
<a href="#">Altering the Access List</a>	5-8
<a href="#">Altering the LANMAN Process</a>	5-9
<a href="#">Aborting the LANMAN Process</a>	5-9
<a href="#">Replacing a LANMON Without a Cold Load</a>	5-10

### **A. Command Summary**

### **B. SCF Error Messages**

### **Index**

### **Examples**

### **Figures**

<a href="#">Figure 2-1.</a>	<a href="#">Relationship Between SLSA Components, Processors, and Adapters</a>	2-4
<a href="#">Figure 2-2.</a>	<a href="#">Multiple TCP/IP Processes for One DNS Entry</a>	2-6
<a href="#">Figure 2-3.</a>	<a href="#">Single TCP/IP Process for One DNS Entry</a>	2-7
<a href="#">Figure 2-4.</a>	<a href="#">The SLSA Subsystem Role Within a NonStop System</a>	2-8
<a href="#">Figure 2-5.</a>	<a href="#">SCF Interface to the SLSA Subsystem</a>	2-10
<a href="#">Figure 2-6.</a>	<a href="#">ATM Object Hierarchy</a>	2-14
<a href="#">Figure 2-7.</a>	<a href="#">Expand and ATMSAP</a>	2-15
<a href="#">Figure 3-1.</a>	<a href="#">Example of a Completed Processor Multifunction (PMF) CRU Configuration Form (For NonStop S-Series Servers Only)</a>	3-10
<a href="#">Figure 3-2.</a>	<a href="#">Example of a Completed Gigabit Ethernet 4-Port ServerNet Adapter (G4SA) Configuration Form</a>	3-11
<a href="#">Figure 4-1.</a>	<a href="#">SCF Object Hierarchy</a>	4-5
<a href="#">Figure 4-2.</a>	<a href="#">Expand and ATMSAP</a>	4-23

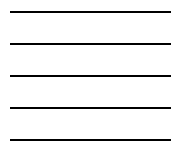
### **Tables**

<a href="#">Table i.</a>	<a href="#">Summary of Contents</a>	xi
<a href="#">Table 2-1.</a>	<a href="#">SLSA Software Components</a>	2-2
<a href="#">Table 2-2.</a>	<a href="#">SLSA Data Structures in Adapters</a>	2-3
<a href="#">Table 3-1.</a>	<a href="#">Naming Convention for SLSA Subsystem and Related Processes (G06.26 and Earlier RVUs)</a>	3-2

**Tables** (continued)

<a href="#"><u>Table 3-2.</u></a>	<a href="#"><u>Naming Convention for SLSA Subsystem and Related Processes (G06.27 and Later G-Series RVUs, and H06.03 and Later H-Series RVUs, and J06.03 and Later J-Series RVUs)</u></a>	3-6
<a href="#"><u>Table 4-1.</u></a>	<a href="#"><u>SCF Commands and Objects</u></a>	4-2
<a href="#"><u>Table 4-2.</u></a>	<a href="#"><u>Sensitive and Nonsensitive SCF Commands</u></a>	4-4
<a href="#"><u>Table 4-3.</u></a>	<a href="#"><u>Operational States for SLSA Objects</u></a>	4-6
<a href="#"><u>Table 4-4.</u></a>	<a href="#"><u>select-spec for a MONITOR object</u></a>	4-137
<a href="#"><u>Table 4-5.</u></a>	<a href="#"><u>select-spec for a PIF object</u></a>	4-141
<a href="#"><u>Table 4-6.</u></a>	<a href="#"><u>select-spec for a SAC object</u></a>	4-147





# What's New in This Manual

## Manual Information

### Abstract

This manual describes how to configure, operate, and manage the ServerNet LAN Systems Access (SLSA) subsystem on an HP Integrity Nonstop™ NS-series or HP NonStop S-series server. This manual includes detailed descriptions of the Subsystem Control Facility (SCF) commands used with the SLSA subsystem and a quick-reference section showing SCF command syntax.

### Product Version

SLSA G06

### Supported Release Version Updates (RVUs)

This manual supports G06.24 and all subsequent G-series RVUs and H06.03 and all subsequent H-series RVUs and J06.03 and all subsequent J-series RVUs until otherwise indicated by its replacement edition.

Part Number	Published
520469-011	August 2010

### Document History

Part Number	Product Version	Published
520469-007	SLSA G06	July 2005
520469-008	SLSA G06	September 2005
520469-009	SLSA G06	May 2006
520469-010	SLSA G06	November 2006
520469-011	SLSA G06	August 2010

# New and Changed Information

## New and Changed Information for the 520469-11 Edition

This edition of the manual has been updated for Token-Ring connectivity on NonStop BladeSystems and includes these updates:

- [ServerNet Architecture and SLSA](#) on page 2-5
- [SLSA Subsystem Management](#) on page 2-10
- [Supported Adapters by NonStop System Type](#) on page 2-13
- [HP Manufacturing Naming Conventions \(For G06.27 and Later G-Series RVUs and HO6.03 and Later H-Series RVUs and J06.03 and Later J-Series RVUs\)](#) on page 3-6
- [Adding an Adapter to the System](#) on page 5-1

## New and Changed Information for the 520469-10 Edition

This edition of the manual has been updated for the Versatile I/O (VIO) enclosure and includes these updates:

- The list of related manuals has changed. See [Where to Get More Information](#) on page xii.
- Changes have been made to the overview information. See [Overview of SLSA](#) on page 2-1 and [Adapters](#) on page 2-12.
- Changes have been made to [HP Manufacturing Naming Conventions \(For G06.27 and Later G-Series RVUs and HO6.03 and Later H-Series RVUs and J06.03 and Later J-Series RVUs\)](#) on page 3-6 and [Suggested Naming Conventions for LIFs](#) on page 3-8.
- An example has been added to [Configuration File](#) on page 3-12.
- Changes have been made to the [ADD ADAPTER Command](#) on page 4-15.
- Changes have been made to [Adding an Adapter to the System](#) on page 5-1.
- The glossary has been removed. For a definition of terms, see the *NonStop System Glossary* in the NonStop Technical Library (NTL.)

## New and Changed Information for the 520469-009 Edition

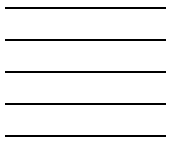
The *LAN Configuration and Management Manual* has been revised to include:

- Updated physical slot information for the Gigabit Ethernet 4-port ServerNet adapter (G4SA) (See [slot](#) on page 3-7.)
- Updated reference to planning guides (See [LAN Configuration](#) on page 3-9.)

# HP Encourages Your Comments

HP encourages your comments concerning this document. We are committed to providing documentation that meets your needs. Send any errors found, suggestions for improvement, or compliments to [docsfeedback@hp.com](mailto:docsfeedback@hp.com). Include the document title, part number, and any comment, error found, or suggestion for improvement concerning this document.





# About This Manual

This manual describes how to configure, operate, and manage the ServerNet LAN Systems Access (SLSA) subsystem on an Integrity NonStop NS-series or NonStop S-series server. This manual includes descriptions of the Subsystem Control Facility (SCF) commands used with the SLSA subsystem and a quick-reference section showing the command syntax.

## Who Should Use This Manual

This manual is written for anyone who configures, manages, or monitors the SLSA subsystem on an Integrity NonStop NS-series or NonStop S-series server.

## What’s in This Manual

---

**Note.** The Event messages (EMS) now reside in the *Operator Messages Manual*.

---

**Table i. Summary of Contents**

Section	Title	This section. . .
1	<a href="#">Configuration Quick Start</a>	Provides key tasks for configuring your system quickly.
2	<a href="#">Introduction to the SLSA Subsystem</a>	Describes the components and fault-tolerance of the ServerNet LAN Systems Access (SLSA) subsystem.
3	<a href="#">SLSA Subsystem Installation and Configuration</a>	Provides installation and configuration information for the SLSA subsystem and describes current configuration restrictions.
4	<a href="#">SCF Commands for the SLSA Subsystem</a>	Provides a detailed description and the complete syntax of the Subsystem Control Facility (SCF) commands applicable to the SLSA subsystem and provides examples of how to use those commands.
5	<a href="#">Managing the SLSA Subsystem</a>	Provides descriptions of procedures for managing the SLSA subsystem and how the SLSA subsystem can recover from various types of failures.
A	<a href="#">Command Summary</a>	Provides the SCF command syntax for SLSA.
B	<a href="#">SCF Error Messages</a>	Provides the text, cause, effect, and recovery for the SLSA SCF error messages.

This manual also contains a glossary of technical terms and abbreviations used throughout the text.

# Where to Get More Information

Depending on the tasks you are performing, you might need the following manuals:

- *NonStop NS-Series Planning Guide*
- *NonStop NS16000 Planning Guide* (available in H06.08 and subsequent H-series RVUs)
- *NonStop NS14000 Planning Guide* (available in H06.08 and subsequent H-series RVUs)
- *NonStop NS1000 Planning Guide* (available in H06.08 and subsequent H-series RVUs)
- *NonStop S-Series Planning and Configuration Guide*
- *Versatile I/O Manual*
- *TCP/IP (Parallel Library) Configuration and Management Manual*
- *TCP/IP Configuration and Management Manual*
- *TCP/IPv6 Configuration and Management Manual*
- *QIO Configuration and Management Manual*
- *PAM Configuration and Management Manual*

## Notation Conventions

### Hypertext Links

Blue underline is used to indicate a hypertext link within text. By clicking a passage of text with a blue underline, you are taken to the location described. For example:

This requirement is described under [Backup DAM Volumes and Physical Disk Drives](#) on page 3-2.

### General Syntax Notation

This list summarizes the notation conventions for syntax presentation in this manual.

**UPPERCASE LETTERS.** Uppercase letters indicate keywords and reserved words. Type these items exactly as shown. Items not enclosed in brackets are required. For example:

MAXATTACH

**lowercase italic letters.** Lowercase italic letters indicate variable items that you supply. Items not enclosed in brackets are required. For example:

*file-name*

**computer type.** Computer type letters within text indicate C and Open System Services (OSS) keywords and reserved words. Type these items exactly as shown. Items not enclosed in brackets are required. For example:

```
myfile.c
```

**italic computer type.** *Italic computer type* letters within text indicate C and Open System Services (OSS) variable items that you supply. Items not enclosed in brackets are required. For example:

```
pathname
```

**[ ] Brackets.** Brackets enclose optional syntax items. For example:

```
TERM [ \system-name. ] $terminal-name
INT[ ERRUPTS ]
```

A group of items enclosed in brackets is a list from which you can choose one item or none. The items in the list can be arranged either vertically, with aligned brackets on each side of the list, or horizontally, enclosed in a pair of brackets and separated by vertical lines. For example:

```
FC [  num  ]
   [ -num  ]
   [ text  ]

K [ X | D ] address
```

**{ } Braces.** A group of items enclosed in braces is a list from which you are required to choose one item. The items in the list can be arranged either vertically, with aligned braces on each side of the list, or horizontally, enclosed in a pair of braces and separated by vertical lines. For example:

```
LISTOPENS PROCESS { $appl-mgr-name }
                  { $process-name  }

ALLOWSU { ON | OFF }
```

**| Vertical Line.** A vertical line separates alternatives in a horizontal list that is enclosed in brackets or braces. For example:

```
INSPECT { OFF | ON | SAVEABEND }
```

**... Ellipsis.** An ellipsis immediately following a pair of brackets or braces indicates that you can repeat the enclosed sequence of syntax items any number of times. For example:

```
M address [ , new-value ]...
[ - ] { 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 }...
```

An ellipsis immediately following a single syntax item indicates that you can repeat that syntax item any number of times. For example:

```
"s-char..."
```

**Punctuation.** Parentheses, commas, semicolons, and other symbols not previously described must be typed as shown. For example:

```
error := NEXTFILENAME ( file-name ) ;
LISTOPENS SU $process-name.#su-name
```

Quotation marks around a symbol such as a bracket or brace indicate the symbol is a required character that you must type as shown. For example:

```
"[ repetition-constant-list "]"
```

**Item Spacing.** Spaces shown between items are required unless one of the items is a punctuation symbol such as a parenthesis or a comma. For example:

```
CALL STEPMOM ( process-id ) ;
```

If there is no space between two items, spaces are not permitted. In this example, no spaces are permitted between the period and any other items:

```
$process-name.#su-name
```

**Line Spacing.** If the syntax of a command is too long to fit on a single line, each continuation line is indented three spaces and is separated from the preceding line by a blank line. This spacing distinguishes items in a continuation line from items in a vertical list of selections. For example:

```
ALTER [ / OUT file-spec / ] LINE
      [ , attribute-spec ]...
```

**!i and !o.** In procedure calls, the !i notation follows an input parameter (one that passes data to the called procedure); the !o notation follows an output parameter (one that returns data to the calling program). For example:

```
CALL CHECKRESIZESEGMENT ( segment-id                !i
                        , error                      !o
                        ) ;
```

**!i,o.** In procedure calls, the !i,o notation follows an input/output parameter (one that both passes data to the called procedure and returns data to the calling program). For example:

```
error := COMPRESSEDIT ( filenum ) ;                !i,o
```

**!i:i.** In procedure calls, the !i:i notation follows an input string parameter that has a corresponding parameter specifying the length of the string in bytes. For example:

```
error := FILENAME_COMPARE_ ( filename1:length        !i:i
                           , filename2:length ) ;    !i:i
```



**!o:i.** In procedure calls, the !o:i notation follows an output buffer parameter that has a corresponding input parameter specifying the maximum length of the output buffer in bytes. For example:

```
error := FILE_GETINFO_ (  filename                                !i
                        , [ filename:maxlen ] ) ;                !o:i
```

## Notation for Messages

This list summarizes the notation conventions for the presentation of displayed messages in this manual.

**Bold Text.** Bold text in an example indicates user input typed at the terminal. For example:

```
ENTER RUN CODE
?123
CODE RECEIVED:      123.00
```

The user must press the Return key after typing the input.

**Nonitalic text.** Nonitalic letters, numbers, and punctuation indicate text that is displayed or returned exactly as shown. For example:

```
Backup Up.
```

**lowercase italic letters.** Lowercase italic letters indicate variable items whose values are displayed or returned. For example:

```
p-register
process-name
```

**[ ] Brackets.** Brackets enclose items that are sometimes, but not always, displayed. For example:

```
Event number = number [ Subject = first-subject-value ]
```

A group of items enclosed in brackets is a list of all possible items that can be displayed, of which one or none might actually be displayed. The items in the list can be arranged either vertically, with aligned brackets on each side of the list, or horizontally, enclosed in a pair of brackets and separated by vertical lines. For example:

```
proc-name trapped [ in SQL | in SQL file system ]
```

**{ } Braces.** A group of items enclosed in braces is a list of all possible items that can be displayed, of which one is actually displayed. The items in the list can be arranged either vertically, with aligned braces on each side of the list, or horizontally, enclosed in a pair of braces and separated by vertical lines. For example:

```
obj-type obj-name state changed to state, caused by
{ Object | Operator | Service }

process-name State changed from old-objstate to objstate
{ Operator Request. }
{ Unknown. }
```

**| Vertical Line.** A vertical line separates alternatives in a horizontal list that is enclosed in brackets or braces. For example:

```
Transfer status: { OK | Failed }
```

**% Percent Sign.** A percent sign precedes a number that is not in decimal notation. The % notation precedes an octal number. The %B notation precedes a binary number. The %H notation precedes a hexadecimal number. For example:

```
%005400
%B101111
%H2F
P=%p-register E=%e-register
```

## Notation for Management Programming Interfaces

This list summarizes the notation conventions used in the boxed descriptions of programmatic commands, event messages, and error lists in this manual.

**UPPERCASE LETTERS.** Uppercase letters indicate names from definition files. Type these names exactly as shown. For example:

```
ZCOM-TKN-SUBJ-SERV
```

**lowercase letters.** Words in lowercase letters are words that are part of the notation, including Data Definition Language (DDL) keywords. For example:

```
token-type
```

**!r.** The !r notation following a token or field name indicates that the token or field is required. For example:

```
ZCOM-TKN-OBJNAME          token-type ZSPI-TYP-STRING.          !r
```

**!o.** The !o notation following a token or field name indicates that the token or field is optional. For example:

```
ZSPI-TKN-MANAGER          token-type ZSPI-TYP-FNAME32.          !o
```

## Change Bar Notation

Change bars are used to indicate substantive differences between this manual and its preceding version. Change bars are vertical rules placed in the right margin of changed portions of text, figures, tables, examples, and so on. Change bars highlight new or revised information. For example:

The message types specified in the REPORT clause are different in the COBOL environment and the Common Run-Time Environment (CRE).

The CRE has many new message types and some new message type codes for old message types. In the CRE, the message type SYSTEM includes all messages except LOGICAL-CLOSE and LOGICAL-OPEN.

## Abbreviations

**ATM.** Asynchronous Transfer Mode

**ATM3SA.** ATM 3 ServerNet adapter

**CCSA.** Common Communication ServerNet adapter

**CRU.** customer-replaceable unit

**DIH.** driver interrupt handler

**DSM.** Distributed Systems Management

**E4SA.** Ethernet 4 ServerNet adapter

**EMS.** Event Management Service

**FCSA.** Fibre Channel ServerNet adapter

**FESA.** Fast Ethernet ServerNet adapter

**FRU.** field-replaceable unit

**GESA.** Gigabit Ethernet ServerNet adapter

**G4SA.** Gigabit Ethernet 4-port ServerNet adapter

**IOAM.** I/O adapter module

**IPX/SPX.** Internet Packet Exchange/Sequenced Packet Exchange

**I/O.** input/output

**IOMF CRU.** I/O multifunction customer replaceable unit

**LAN.** local area network

**LANMAN.** LAN Manager

**LANMON.** LAN Monitor

**LIF.** logical interface

**MAC.** media access control

**mbps.** megabits per second

**MAU.** medium attachment unit

**MFIOB.** multifunction I/O board

**PIF.** physical interface

**PMF CRU.** processor multifunction customer replaceable unit

**POST.** power-on self-test

**PVC.** Permanent Virtual Circuit

**QIO.** Queued input output

**SAC.** servernet addressable controller

**SAN.** system area network

**SCF.** Subsystem Control Facility

**SCP.** Subsystem Control Point

**SEB.** ServerNet expansion board

**SMB.** serial maintenance bus

**SLSA.** ServerNet LAN Systems Access

**SP.** service processor

**TCP/IP.** Transmission Control Protocol/Internet Protocol

**TRSA.** Token Ring ServerNet adapter

**SWAN.** ServerNet wide area network

**VCC.** Virtual Channel Connection

**WAN.** wide area network

**WAN IOP.** wide area network input/output process

**VIO.** Versatile I/O

# 1 Configuration Quick Start

This section describes your initial ServerNet LAN Systems Access (SLSA) subsystem configuration and how to verify that required SLSA subsystem processes and objects are configured and started.

---

**Note.** All adapters ordered with a NonStop system are configured by HP manufacturing in the initial system configuration. Only follow the procedures described in this section if you are altering the configuration of the SLSA subsystem provided by HP or are installing additional adapters.

---

<a href="#">Initial SLSA Subsystem Configuration</a>	<a href="#">1-1</a>
<a href="#">Verifying the Initial SLSA Subsystem Configuration</a>	<a href="#">1-2</a>
<a href="#">Troubleshooting Tips</a>	<a href="#">1-5</a>
<a href="#">Installing an Adapter</a>	<a href="#">1-7</a>
<a href="#">Changing the Initial SLSA Subsystem Configuration</a>	<a href="#">1-8</a>

To become familiar with the SLSA subsystem, refer to [Section 2, Introduction to the SLSA Subsystem](#).

## Initial SLSA Subsystem Configuration

This subsection describes how the SLSA subsystem is configured by HP manufacturing and includes information about:

- The contents of the initial configuration files
- Verification and troubleshooting tips for your initial subsystem configuration
- ServerNet adapters used by the SLSA subsystem:
  - ATM 3 ServerNet adapters (ATM3SAs)
  - 6763 Common Communications ServerNet adapters (CCSAs)
  - Ethernet 4 ServerNet adapter (E4SA)
  - Fast Ethernet ServerNet adapter (FESA)
  - Gigabit Ethernet ServerNet adapter (GESA)
  - Gigabit Ethernet 4-port ServerNet adapter (G4SA)
  - Token-Ring ServerNet adapter (TRSA)
- Versatile I/O (VIO) enclosure in NonStop NS-series servers

## Configuration File Contents

HP manufacturing used the following configuration files to create your initial system configuration. These files are located in the \$SYSTEM.ZSYSCONF subvolume.

- SCF0000
- STARTCOM
- STARTSCF

### Contents of the SCF0000 File

The SCF0000 file contains Subsystem Control Facility (SCF) commands that configure the following LAN subsystem processes and objects:

- LAN Manager (LANMAN) process (\$ZZLAN).
- ADAPTER and logical interface (LIF) objects.

The SCF000 file also contains SCF commands that configure and start the SWAN concentrators connected to the LAN adapters.

### Contents of the STARTCOM File

The STARTCOM file contains HP Tandem Advanced Command Language (TACL) commands that configure the HP NonStop TCP/IP processes (and corresponding LISTNER and TELSERV processes) used to support the adapters defined in the SCF0000 file.

### Contents of the STARTSCF File

The STARTSCF file contains SCF commands that configure and start the TCP/IP SUBNET and ROUTE objects for the NonStop TCP/IP processes defined in the STARTCOM file. These commands associate each adapter LIF to a TCP/IP process.

The TCP/IP processes for the LIFs associated with the multi-function I/O board (MFIOB) adapters installed in a NonStop S-series server are configured and started by the OSM Service Connection or TSM Service Application. The OSM or TSM processes are added in the SCF0000 file.

## Verifying the Initial SLSA Subsystem Configuration

HP recommends that you verify that the required processes and objects are configured and started. The following subsections describe how to use the SCF commands to check the state of these processes and objects. All the processes must be in the STARTED state. If a process is in the STOPPED state, refer to [Troubleshooting Tips](#).

## Checking the State of the LAN Manager (LANMAN) Process

Use the SCF STATUS PROCESS command to determine the current state of the LAN Manager (LANMAN) process:

```
STATUS PROCESS $ZZLAN
```

## Checking the State of the LAN Monitor (LANMON) Processes

**Note.** As of the G06.21 RVU, a new LANMON can be installed without a cold load. Refer to [Replacing a LANMON Without a Cold Load](#) on page 5-10 for more information.

Use the SCF STATUS MON command to determine the current state of each LAN Monitor (LANMON) process. You must enter the ALLOW ALL ERRORS command before using the wildcard (\*) if your system has fewer than 16 processors.

```
ALLOW ALL ERRORS
```

```
STATUS MON $ZZLAN.#ZLM*
```

## Checking the State of the TCP/IP Processes

Use the appropriate SCF LISTDEV TCPIP command to get a list of all running TCP/IP processes.

For conventional TCP/IP:

```
LISTDEV TCPIP
```

**Note.** Parallel Library TCP/IP is not supported on Integrity NonStop NS-series servers.

For Parallel Library TCP/IP:

```
LISTDEV PTCPIP
```

For NonStop TCP/IPv6:

```
LISTDEV TCPIPv6
```

Use the SCF STATUS PROCESS command to determine the state of each preconfigured TCP/IP process:

```
STATUS PROCESS $tcpip process-name
```

## Checking the State of the Subnets

Use the SCF STATUS SUBNET command to determine the state of each preconfigured subnet:

For conventional TCP/IP:

```
STATUS SUBNET $tcpip process-name.*
```

**Note.** Parallel Library TCP/IP is not supported on Integrity NonStop NS-series servers.

For Parallel Library TCP/IP and NonStop TCP/IPv6:

```
STATUS SUBNET $ZZTCP.*
```

## Checking the State of the Routes

Use the SCF STATUS ROUTE command to determine the state of each preconfigured route:

For conventional TCP/IP:

```
STATUS ROUTE $tcpip process-name.*
```

For Parallel Library TCP/IP and NonStop TCP/IPv6:

```
STATUS ROUTE $ZZTCP.*
```

## Checking the State of the SLSA Subsystem Objects

Use the SCF STATUS ADAPTER command to get a list of all the running adapters in the SLSA subsystem:

```
STATUS ADAPTER $ZZLAN.*
```



# Troubleshooting Tips

All the required processes and objects described in [Initial SLSA Subsystem Configuration](#) should be started during the system-load sequence. If you discover that a process or object is in the STOPPED state, determine the cause of the problem, then start the process or other object.

The following are some general troubleshooting guidelines:

- Examine the contents of the event-message log: a WAN subsystem or Kernel subsystem event message might have been issued that provides information about the process failure.
- Try to start the process or object using the commands described in this subsection or execute the STARTCOM file by entering the following command at the TACL prompt:

```
TACL /IN $SYSTEM.ZSYSCONF.STARTCOM, NAME/
```

The STARTCOM file invokes the STARTSCF file. The contents of both files are described in the *WAN Subsystem Configuration and Management Manual*. You must be logged on as SUPER.SUPER to run the STARTCOM file.

## Starting the LANMAN Process

If the LANMAN process is in the STOPPED state, try to start it by using the following SCF command:

```
START PROCESS $ZZKRN.#ZZLAN
```

The priority of the LANMAN process should stay fairly constant over time; the priority can vary depending on how busy the NonStop NS-series or NonStop S-series server is, but if the priority goes down by a substantial amount, a problem may exist.

The creation time of the LANMAN process should occur shortly after the NonStop server was last started. (You can use the STATUS PROCESS \$ZZKRN.#ZZLAN, DETAIL command from SCF or the STATUS \$ZZLAN, DETAIL command from TACL to determine the process-creation time.)

## Starting a LANMON Process

If a LANMON process is in the STOPPED state, use the SCF START MON command to start it:

```
START MON $ZZLAN.#ZLM processor-id
```

*processor-id*

is the number of the processor in which you want the LANMON to start.

## Starting a TCP/IP Process

△ **Caution.** NonStop TCP/IPv6 and Parallel Library TCP/IP are incompatible and cannot run on the same system. Parallel Library TCP/IP is not supported on Integrity NonStop NS-series servers.

- If you are using conventional TCP/IP and there is no TCP/IP process, rerun the STARTCOM file. STARTCOM configures the TCP/IP processes and calls STARTSCF which starts the subnets and routes.
- If you are using NonStop TCP/IPv6, see the *TCP/IPv6 Configuration and Management Manual* for information about starting a NonStop TCP/IPv6 process.
- If you are using Parallel Library TCP/IP, see the *TCP/IP (Parallel Library) Configuration and Management Manual* for information about starting a Parallel Library TCP/IP process.

## Starting the LAN Subsystem Objects

### Starting an ADAPTER Object

If objects associated with the adapter are in the STOPPED state, use the SCF START command to start all the adapter's subordinate objects: ServerNet addressable controllers (SACs) and physical interfaces (PIFs):

```
START ADAPTER $ZZLAN.adapter-name, SUB ALL
```

*adapter-name*

specifies the adapter to start.

### Starting a SAC Object

If a SAC object is in the STOPPED state, use the SCF START SAC command to start the object and its subordinate PIFs:

```
START SAC $ZZLAN.sac-name, SUB ALL
```

*sac-name*

is the name of the SAC object to start.

## Starting a PIF Object

If a PIF object is in the STOPPED state, use the SCF START PIF command to start it:

```
START PIF $ZZLAN.pif-name
```

*pif-name*

is the name of the PIF object to start.

## Starting a LIF Object

If a LIF object is in the STOPPED state, use the SCF START LIF command to start it:

```
START LIF $ZZLAN.lif-name
```

*lif-name*

is the name of the LIF object to start.

## Starting an ATMSAP Object

```
START ATMSAP $ZZLAN.atmsap-name
```

*atmsap-name*

is the name of the ATMSAP object to start.

# Installing an Adapter

After you verify the initial SLSA subsystem configuration as described on page [1-2](#), you can install additional adapters. For information, see:

### Adapter Manuals (page 1 of 2)

*6763 Common Communication ServerNet Adapter Installation and Support Guide* for the CCSA

*ATM Adapter Installation and Support Guide* for the ATM3SA

*Ethernet Adapter Installation and Support Guide* for the E4SA

*Fast Ethernet Adapter Installation and Support Guide* for the FESA

*Fibre Channel ServerNet Adapter Installation and Support Guide* for the FCSA

*Gigabit Ethernet Adapter Installation and Support Guide* for the GESA

*Gigabit Ethernet 4-Port Adapter Installation and Support Guide* for the G4SA

*NonStop S-Series Hardware Installation and Fast Path Guide* for the MFIOB

*Token-Ring Adapter Installation and Support Guide* for the TRSA

*SWAN Concentrator Installation and Support Guide* for the SWAN Concentrator (T3880)

**Adapter Manuals** (page 2 of 2)

*SWAN 2 Concentrator Installation and Support Guide* for the SWAN 2 Concentrator (T3881)

*Versatile I/O Manual*

*WAN Subsystem Configuration and Management Manual* (SWAN and SWAN 2 concentrator configuration information)

## Changing the Initial SLSA Subsystem Configuration

After installing your adapter, you might want to change the initial SLSA subsystem configuration. For example, you could:

- Assign the IP addresses to different LIF objects
- Add additional adapters
- Reconfigure the TCP/IP processes assigned to the SLSA subsystem

SLSA subsystem management tasks like these are described in [Section 5, Managing the SLSA Subsystem](#).

### WAN Wizard Pro

WAN Wizard Pro provides a graphical user interface (GUI) that guides you through the configuration of wide area network (WAN) and local area network (LAN) hardware and software.

WAN Wizard Pro can be used as an alternative to Subsystem Control Facility (SCF) to configure the following LAN adapters:

ATM 3 ServerNet adapters (ATM3SAs)  
6763 Common Communications ServerNet adapters (CCSAs)  
Ethernet 4 ServerNet adapters (E4SAs)  
Fast Ethernet ServerNet adapters (FESAs)  
Gigabit Ethernet ServerNet adapters (GESAs)  
Gigabit Ethernet 4-port ServerNet adapters (G4SAs)  
Token-Ring ServerNet adapters (TRSAs)

Depending on your RVU, access WAN Wizard Pro from the taskbar on your system console using one of the following methods:

- For G06.21 RVU and later RVUs, including H-series RVUs:  
**Start>Programs>HP WAN Wizard Pro>WAN Wizard Pro**
- For G06.20 RVU and earlier RVUs:  
**Start>Programs>Compaq TSM>Guided Configuration Tools>WAN Wizard Pro**

# Introduction to the SLSA Subsystem

This section describes the ServerNet LAN Systems Access (SLSA) subsystem architecture.

<a href="#">Overview of SLSA</a>	<a href="#">2-1</a>
<a href="#">ServerNet Architecture and SLSA</a>	<a href="#">2-5</a>
<a href="#">The SLSA Role Within the System</a>	<a href="#">2-7</a>
<a href="#">Architecture of the SLSA Subsystem</a>	<a href="#">2-10</a>
<a href="#">Fault Tolerance of the SLSA Subsystem</a>	<a href="#">2-16</a>

## Overview of SLSA

In the NonStop system, the communications adapter communicates with multiple processors simultaneously delivering data directly to the processor that contains the target client.

SLSA operates within the queued input/output (QIO) segment environment of the NonStop system and uses the QIO subsystem to communicate with SLSA clients. Data that meets a filtering criteria is delivered to the QIO queue in the specified processor. SLSA provides that filtering.

SLSA is the interface between NonStop processors and adapters. SLSA gives processors access to the LAN by providing the required input/output (I/O) processes between the LAN service providers and a LAN.

---

**Note.** SLSA supports many different adapters installed in different system types. See [Adapters](#) on page 2-12 or [Adapter Manuals](#) on page 1-7.

---

SLSA LAN clients are processes that use the LAN subsystem to send and receive data on a LAN attached to the NonStop server. Three client interfaces are available: NonStop TCP/IP, PAM, and NonStop IPX/SPX.

SLSA software components reside in the host node and the adapters. [Table 2-1](#) describes these components and [Table 2-2](#) describes the SLSA data structures in the adapters. [Figure 2-1](#) on page 2-4 shows the relationship between the SLSA components, processors, and adapters.

**Table 2-1. SLSA Software Components**

SLSA Software Component	Description
LAN manager (LANMAN) process	<ul style="list-style-type: none"> <li>● Manages all SLSA objects in the system</li> <li>● Receives management commands from the user through SCF and communicates the commands to a LANMON process</li> <li>● Starts and stops the LANMON processes in each processor</li> <li>● Manages the assignment of adapters to specific LANMON processes</li> </ul> <p>For more information about LAN manager, see <a href="#">Processes</a> on page 2-11.</p>
LAN monitor (LANMON) process	<ul style="list-style-type: none"> <li>● Provides trace facilities for all LAN access through a specific processor</li> <li>● Handles events for itself, the driver interrupt handler (DIH), and the adapter</li> <li>● Controls adapters communicating with LANMON by means of DIH</li> <li>● Manages error recovery of adapters, SACs, and PIFs.</li> </ul> <p>For more information about LAN monitor, see <a href="#">Monitors</a> on page 2-12.</p>
driver interrupt handler (DIH)	<p>DIH is a component of the SLSA subsystem that is responsible for the transfer of data and management information and error handling. It is the interface between the rest of the SLSA subsystem and the communications adapter hardware.</p> <p>For more information about DIH, see <a href="#">System-Level Processes and Libraries</a> on page 2-8.</p>
LAN manager library (LMLIB)	<p>LMLIB is a shared library of function calls used for internal purposes only. These calls provide an interface for the SLSA objects. LMLIB is used by the LANMAN and LANMON processes to:</p> <ul style="list-style-type: none"> <li>● Add or delete an object</li> <li>● Retrieve the current state information about an object</li> <li>● Modify the state of an object</li> <li>● Retrieve or modify the attributes of an object</li> </ul>

**Table 2-2. SLSA Data Structures in Adapters**

<b>SLSA Data Structure</b>	<b>Description</b>
Logical interface (LIF)	<p>Allows QIO clients (NonStop TCP/IP and IPX/SPX) to connect their queues to one or more PIFs.</p> <p>For more information about LIFs, see <a href="#">Logical Interface (LIF)</a> on page 2-15.</p>
Physical interface (PIF)	<p>Represents the physical connection to the LAN. The PIF is an abstraction that is used to transmit MAC-layer statistics and information. Data travels from an NSK system to the outside world by using the PIF.</p> <p>For more information about PIFs, see <a href="#">Physical Interface (PIF)</a> on page 2-15.</p>
Servernet addressable controller (SAC)	<p>An object that represents the entry point from the adapter into the ServerNet router. Each SAC has a unique ServerNet address, assigned when the adapter is started. Data from the outside world (symbolized by the PIF) is routed to the correct processor by the SAC.</p> <p>For more information about SACs, see <a href="#">ServerNet Addressable Controller (SAC)</a> on page 2-15.</p>

[Figure 2-1](#) shows that:

- Data from an application is sent by TCP/IP or IPX/SPX into the SLSA QIO segment of the processor.

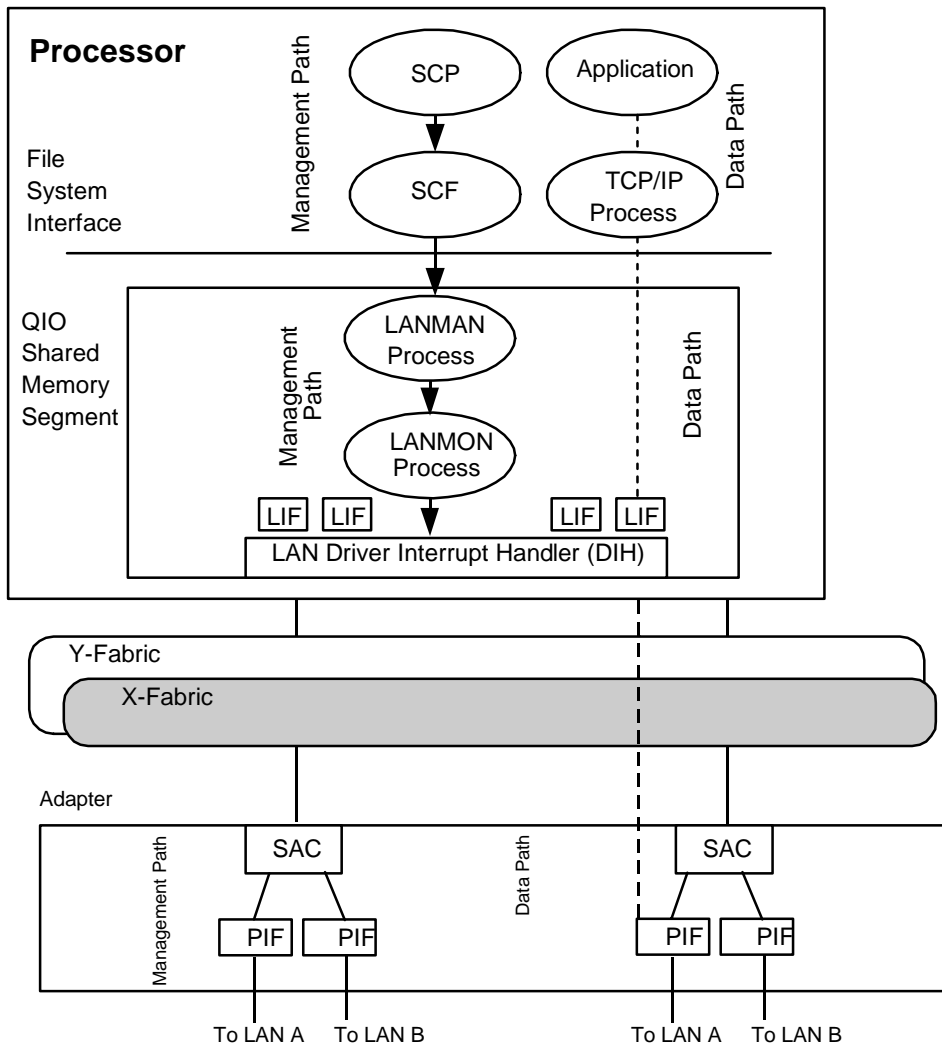
---

**Note.** IPX/SPX is not supported on systems running H-series or J-series RVUs.

---

- From the queue, the data goes to a LIF where the data is associated with one of the PIFs. The data then goes to a SAC, which gives the data the IP address of a LAN.
- From the SAC, the data moves to a PIF and is given a MAC address and sent to the LAN. On the LAN, the data goes to its intended target.

**Figure 2-1. Relationship Between SLSA Components, Processors, and Adapters**



VST805.vsd



# ServerNet Architecture and SLSA

---

**Note.** J-Series RVUs and NonStop BladeSystems have different architecture and networking considerations. For more information, see the *HP NonStop Networking Overview*.

---

The ServerNet architecture on Integrity NonStop NS-series or NonStop S-series servers allows up to 16 processors in a system to have direct access to a logical interface (LIF). This feature allows you to load-balance your data communications with greater ease. For example, you can run Transmission Control Protocol/Internet Protocol (TCP/IP) stacks in multiple processors with access to the same LIF.

Before G05, the TCP/IP process needed to be in the processor that had ownership of the ServerNet addressable controller (SAC). Starting with G05, the SAC ownership concept has changed; the owning process retains responsibility for downloading operational code and for reporting errors but no longer has the exclusive data path to the SAC. Starting with G06, TCP/IP has been modified to take advantage of this new functionality in SLSA.

---

**Note.** Parallel Library TCP/IP is not supported on Integrity NonStop NS-series servers.

---

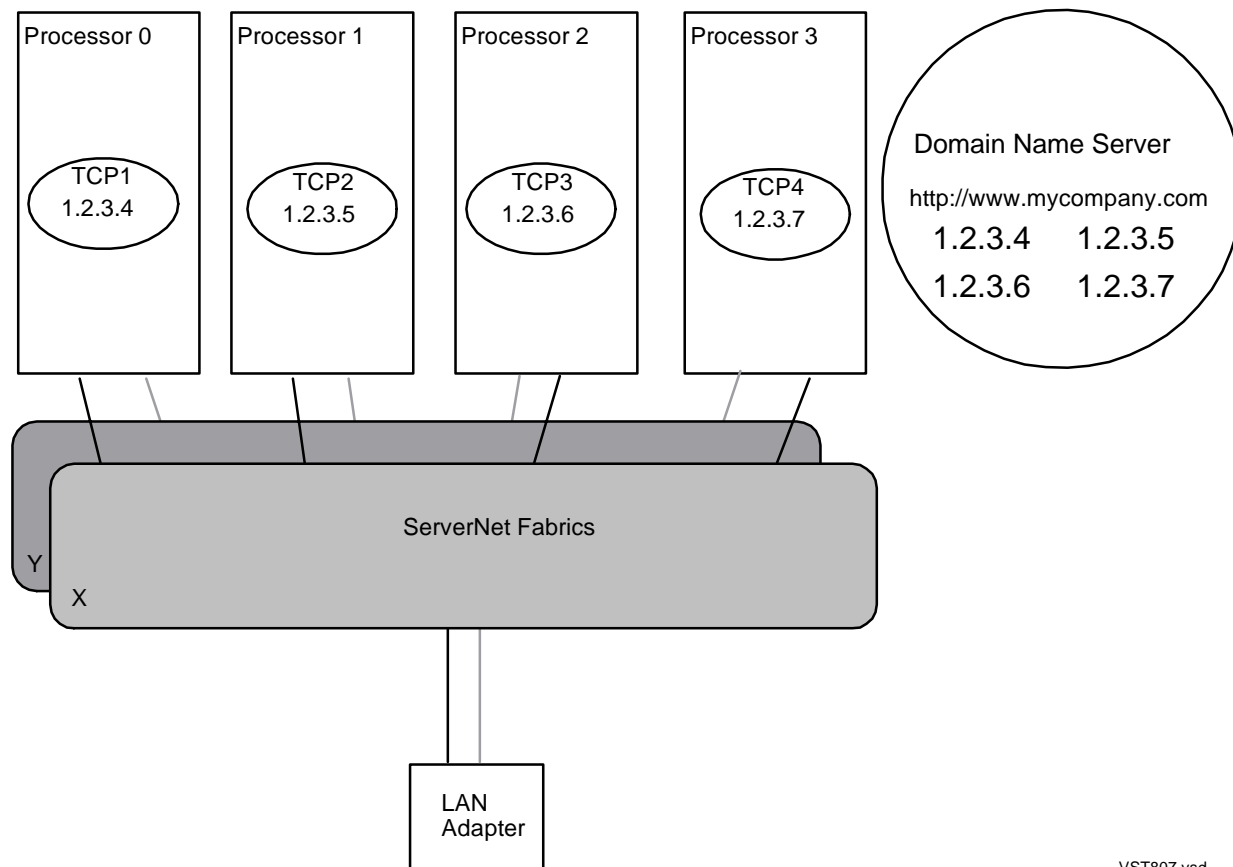
ServerNet architecture and NonStop systems allow you to run a single IP address (provided by Parallel Library TCP/IP and NonStop TCP/IPv6) or multiple IP addresses (provided by multiple TCP/IP processes balanced between multiple processors) over multiple processors that access the same LIF. With Integrity NonStop NS-series and NonStop S-series servers, there is a one-to-many relationship between LIFs and processors.

By contrast, on NonStop K-series servers, multiple IP addresses running in multiple processors require multiple communications controllers (the K-series equivalent of adapters.) There is a one-to-one relationship between communications controllers and processors when supporting multiple IP addresses.

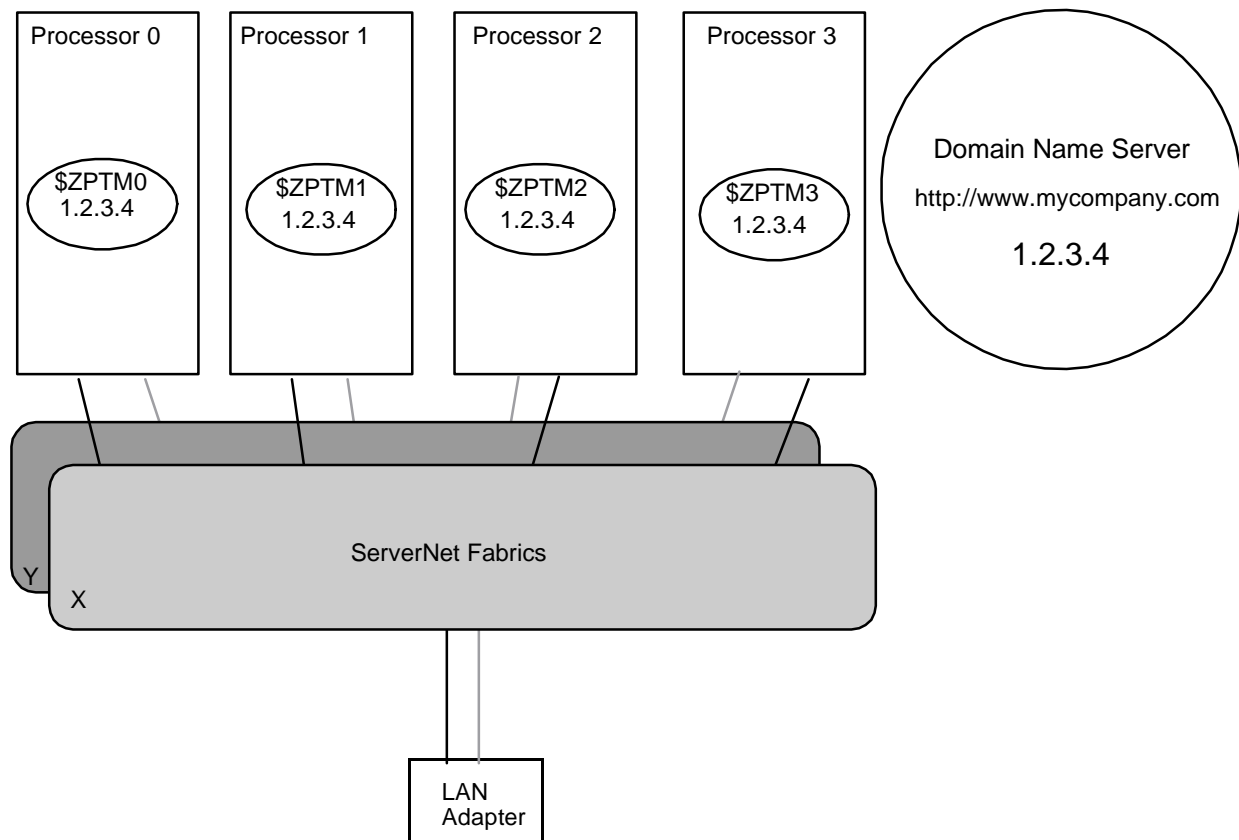
[Figure 2-2](#) on page 2-6 and [Figure 2-3](#) on page 2-7 show examples of an application and a Domain Name Server (DNS) taking advantage of the multiple processor LIF access feature.

In [Figure 2-2](#), the DNS entry has multiple IP addresses for a given name and those addresses can be in different processors supported by a single LIF.

In [Figure 2-3](#), the DNS entry has a single IP address for a given name and the address applies to all processors running Parallel Library TCP/IP or NonStop TCP/IPv6 on a given LIF.

**Figure 2-2. Multiple TCP/IP Processes for One DNS Entry**

VST807.vsd

**Figure 2-3. Single TCP/IP Process for One DNS Entry**

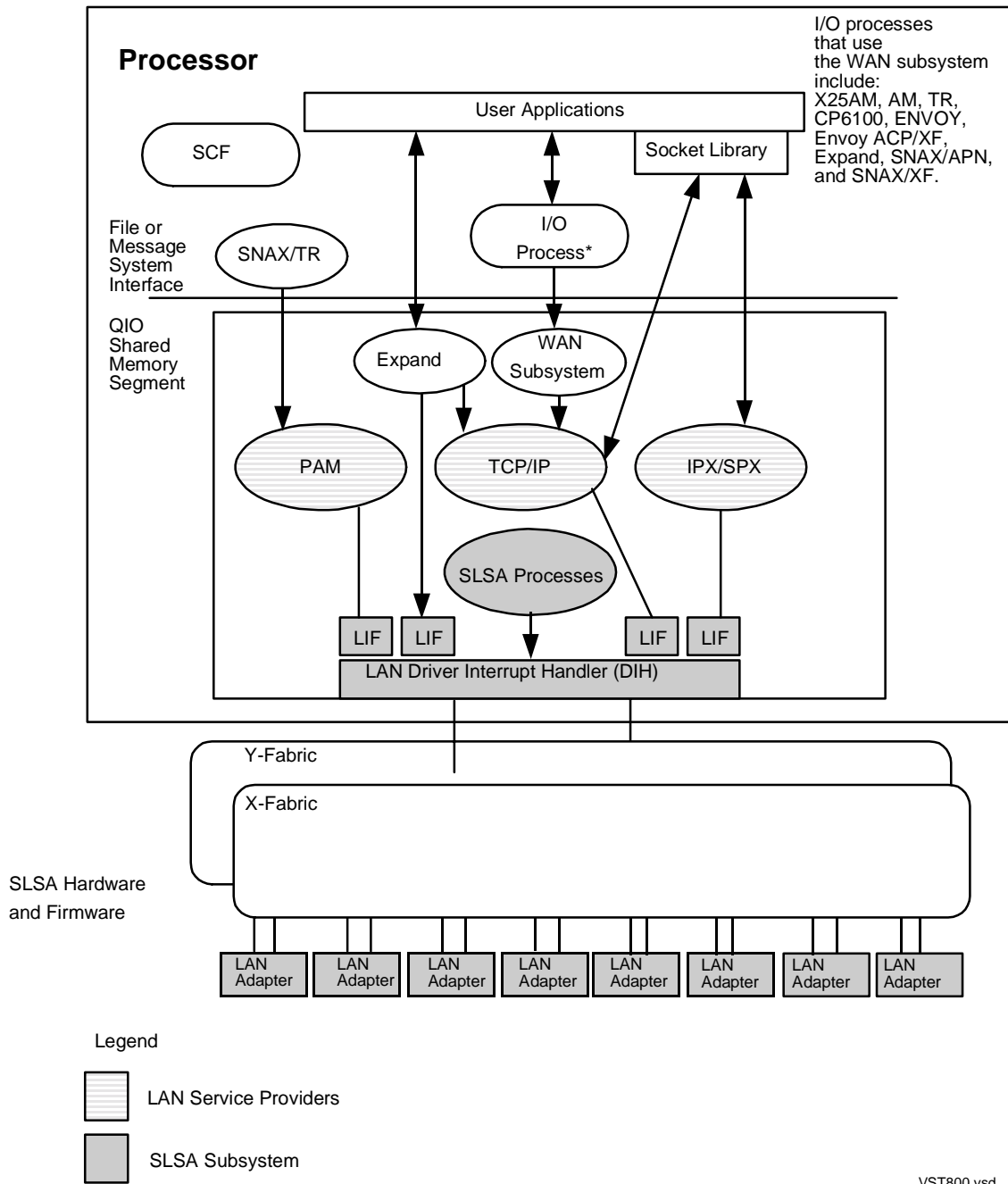
VST808.vsd

## The SLSA Role Within the System

The SLSA subsystem provides access to parallel LAN and WAN I/O for Integrity NonStop NS-series servers and NonStop S-series servers. Data-communications services that use SLSA include the following:

- [System-Level Processes and Libraries](#) on page 2-8
- [LAN Service Providers](#) on page 2-9
- [LAN Clients](#) on page 2-9

[Figure 2-4](#) on page 2-8 shows the role of SLSA within the system.

**Figure 2-4. The SLSA Subsystem Role Within a NonStop System**

## System-Level Processes and Libraries

The SLSA subsystem uses the shared-memory segment (provided by the QIO subsystem) to move the ownership of data between processes. (Refer to the *QIO Configuration and Management Manual* for more information about QIO.) SLSA also uses a QIO-based driver interrupt handler (DIH) to communicate with the appropriate adapter type over the ServerNet system area network (SAN). The DIH controls the

ServerNet adapters and uses system-library routines accessible to LAN service-provider processes (such as TCP/IP).

## LAN Service Providers

Local area network (LAN) service providers are processes that use the SLSA subsystem to send and receive data on a LAN attached to the system. Four directly attached LAN service providers are supported: HP NonStop TCP/IP (includes conventional TCP/IP, Parallel Library TCP/IP, and NonStop TCP/IPv6), HP NonStop Expand, NonStop IPX/SPX, and PAM, which provide access for the following:

QIO API clients	<p>Protocol subsystems that communicate with a LAN-transport service provider directly through QIO. These clients are:</p> <ul style="list-style-type: none"> <li>● Expand products connecting through the QIO application program interface (API) to TCP/IP to provide Expand/IP capability.</li> <li>● WAN input/output process (WAN I/O process) drivers connecting through the QIO API to TCP/IP to provide access to remote ServerNet wide area network (SWAN) concentrators.</li> </ul>
Socket library applications	User applications and HP protocols (such as TELSERV and FTP) that use the socket library to establish remote connections and communicate through TCP/IP and IPX/SPX.
SNAX	SNAX applications running over the token-ring LAN.
Expand	Expand traffic over ATM.

## LAN Clients

LAN clients are processes, user applications, or subsystems that use the SLSA subsystem and related LAN service providers to connect to the Ethernet or token-ring LANs attached to an Integrity NonStop NS-series or NonStop S-series server. In addition, the WAN subsystem is a client of the SLSA subsystem because the WAN subsystem uses the SLSA Ethernet LAN to access the SWAN concentrator.

# Architecture of the SLSA Subsystem

The SLSA subsystem includes hardware components and software processes. SCF provides the management interface for the SLSA subsystem. The elements of the SLSA subsystem architecture are:

- [SLSA Subsystem Management](#)
- [SLSA Subsystem SCF Objects](#) on page 2-11
  - [Processes](#) on page 2-11
  - [Monitors](#) on page 2-12
  - [Adapters](#) on page 2-12
  - [ATMSAP Object](#) on page 2-14
  - [Logical Interface \(LIF\)](#) on page 2-15
  - [Physical Interface \(PIF\)](#) on page 2-15
  - [ServerNet Addressable Controller \(SAC\)](#) on page 2-15

## SLSA Subsystem Management

---

**Note.** Programmatic management of the SLSA subsystem through the Subsystem Programmatic Interface (SPI) is not supported.

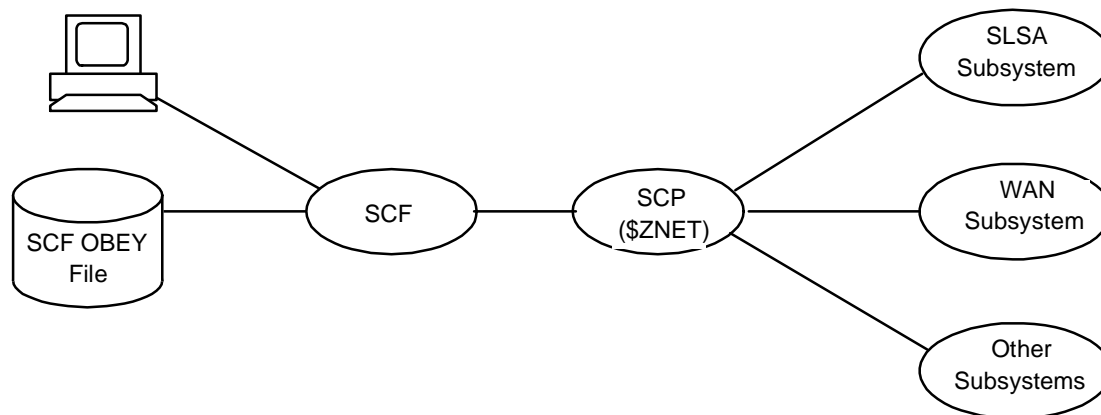
---

The Subsystem Control Facility (SCF) ([Figure 2-5](#)) is the interface used to manage the processes and objects in the SLSA subsystem. For more information about SCF, refer to the *SCF Reference Manual for G-Series RVUs* or *SCF Reference Manual for J-Series and H-Series RVUS*. For a description of the SCF commands, object hierarchy, and object states for the SLSA subsystem, see [Section 4, SCF Commands for the SLSA Subsystem](#).

---

**Figure 2-5. SCF Interface to the SLSA Subsystem**

---



VST801.vsd

## SLSA Subsystem SCF Objects

The SCF objects for the SLSA subsystem consist of manager and monitor processes and several other objects that correspond to components of the adapters that connect the LANs to the server.

The PROCESS and MONITOR objects correspond to the process used to manage (LANMAN) and monitor (LANMON) the SLSA subsystem through SCF.

The SCF objects related to adapters are the ADAPTER, SAC, LIF, and PIF objects. The SAC and PIF objects correspond directly to the adapter's hardware. The LIF objects correspond to logical processes running on the system that are used for handling data transferred between the LAN and a system using the ServerNet architecture. The LIF provides an interface to the PIF. [Figure 2-4](#) on page 2-8 shows the relationship between the SLSA SCF objects as they correspond to an Ethernet adapter. (The dashed lines in [Figure 2-4](#) indicate the data path.)

### Processes

The SLSA subsystem processes manage and control the SLSA subsystem. The manager process is the LAN Manager (LANMAN) process.

The LANMAN process is the central point of management for all LAN connectivity in an Integrity NonStop NS-series or NonStop S-series server. The LANMAN process runs as a process pair in processors 0 and 1 only (the same processors as the system disk). LANMAN performs the following tasks:

- Accepts all Distributed Systems Management (DSM) requests from the SCF product module through the Subsystem Control Point (SCP).
- Starts a LAN Monitor (LANMON) process in each processor and maintains it persistently across failures.
- Assigns ownership of ServerNet addressable controllers (SACs) to specific LANMON processes for management purposes.

LANMAN begins running as a persistent generic process at system startup with the name \$ZZLAN.

## Monitors

---

**Note.** As of the G06.21 RVU, a new LANMON can be installed without a cold load. Refer to [Replacing a LANMON Without a Cold Load](#) on page 5-10 for more information.

---

The monitor objects are the control processes in the SLSA subsystem. The LANMAN process creates a monitor in each processor. If a monitor process fails, LANMAN restarts it. Each LAN monitor (LANMON) performs the following functions:

- Controls the access to specific adapters by:
  - Creating data structures representing all the adapters, SACs, physical interfaces (PIFs), and logical interfaces (LIFs).
  - Handling all error reporting from the driver interrupt handlers (DIHs).
  - Providing trace facilities for all LAN access from its processor.
- Broadcasts state changes for each SAC for which it has access to the other LANMON processes in the system.

LANMON names have the form \$ZZLAN.#ZLM $nn$ , where  $nn$  indicates the particular processor in which the LANMON is running. For example, \$ZZLAN.#ZLM01 indicates the LANMON process running in processor 01 of the system.

## Adapters

---

**Note.** J-Series RVUs and NonStop BladeSystems have different architecture and networking considerations. For more information, see the *HP NonStop Networking Overview* or the *NonStop BladeSystem Planning Guide*.

---

Ethernet adapters and their ports are the primary hardware components that provide a NonStop S-series or NonStop NS-series system LAN access. The Ethernet ports support one or more SACs. Adapters are installed in a system enclosure or IOAM enclosure and connect to the enclosure through the ServerNet SAN. In a Versatile I/O (VIO) enclosure, Ethernet ports in slots 6 or 7 function like G4SAs and are treated as G4SAs by the Integrity NonStop NS-series system. (The VIO enclosure is available in H06.08 and later H-series RVUs.) For more information about the VIO enclosure, see the *Versatile I/O Manual*.

Adapter names use the form \$ZZLAN.*adapter-name*, where *adapter-name* consists of up to eight alphanumeric characters with a leading alphabetic character, for example: \$ZZLAN.MIOE0 or \$ZZLAN.G4SA1. (See [HP Manufacturing Naming Conventions \(G06.26 and Earlier RVUs\)](#) on page 3-2 for information about preconfigured naming conventions and suggestions for naming your adapters.)



## Supported Adapters by NonStop System Type

- [Adapters for NonStop S-Series systems](#)
- [Adapters for NonStop NS-Series](#)
- [Adapters for NonStop BladeSystems \(J06.03 and Later RVUs\)](#)

### Adapters for NonStop S-Series systems

- Multifunction I/O Board/Ethernet (MFIOB)
- ATM ServerNet Adapter (ATM3SA)
- Ethernet 4 ServerNet adapter (E4SA)
- Fast Ethernet ServerNet adapter (FESA)
- Gigabit Ethernet ServerNet adapter (GESA)
- 6763 Common Communication ServerNet Adapter (CCSA)
- Token-Ring ServerNet adapter (TRSA)
- Gigabit Ethernet 4-port ServerNet adapter (G4SA)

---

**Note.** G4SAs interoperate with NonStop S-series systems but cannot be directly installed in an S-series enclosure.

---

### Adapters for NonStop NS-Series

- G4SAs are the only adapter that can be directly installed in a NonStop NS-series system's IOAM enclosure.
- [Adapters for NonStop S-Series systems](#) can also connect to NonStop NS-series systems if your system model supports these connections. To determine if your NS-series system supports connections to S-series I/O enclosures, see your installation documentation (for example, the *NonStop NS16000 Hardware Installation Manual*).

### Adapters for NonStop BladeSystems (J06.03 and Later RVUs)

Three ServerNet adapters are supported on NonStop BladeSystems:

- **Gigabit Ethernet 4-port ServerNet adapter (G4SA)**  
G4SAs can only be installed in an IOAM. Up to 4 IOAMs can be installed in a NonStop BladeSystem enclosure.
- **6763 Common Communication ServerNet Adapter (CCSA)**  
CCSA-2 adapters in a NonStop S-series I/O enclosure provide SS7 connectivity to NonStop BladeSystems via IOMF2 CRU to c7000 ServerNet switch connections.
- **Token-Ring ServerNet adapter (TRSA)**  
TRSAs in a NonStop S-series I/O enclosure provide token-ring connectivity to NonStop BladeSystems via IOMF2 CRU to c7000 ServerNet switch connections.

For information about MFIOBs, see the *NonStop S-Series Hardware Installation and Fast Path Guide*.

For information about the other adapters, see [Adapter Manuals](#) on page 1-7.

For information about the NonStop BladeSystem, including its associated hardware components such as the IP CLuster I/O Module (CLIM), which is also used for Ethernet connectivity on J-series RVUs, see the *NonStop BladeSystem Planning Guide*.

## ATMSAP Object

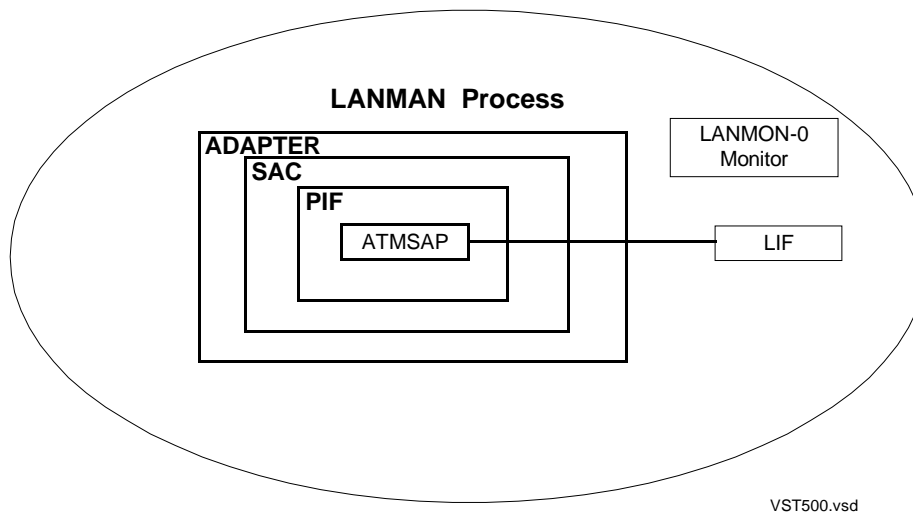
The SLSA ATMSAP object represents an ATM-protocol-direct service access point interface. It provides direct access to a VCC for Expand.

An ATMSAP object can be configured to interface to a permanent virtual circuit (PVC) object. A LIF object must be associated with the ATMSAP object in order to provide host access to the ATMSAP object. A PVC connection provides a static permanent virtual circuit. A PVC is defined with a VCC attribute for the ATMSAP. The VCC is comprised of a VPI, VCI pair.

The ATMSAP object is managed by the ABORT, ADD, ALTER, DELETE, INFO, NAMES, START, STATS, STATUS, and STOP commands.

[Figure 2-6](#) shows the hierarchy of ATM objects. ATMSAP objects are subordinate to PIFs. LIF objects are associated with either a PIF object or an ATMSAP object. No objects can be configured subordinate to an ATMSAP object.

**Figure 2-6. ATM Object Hierarchy**



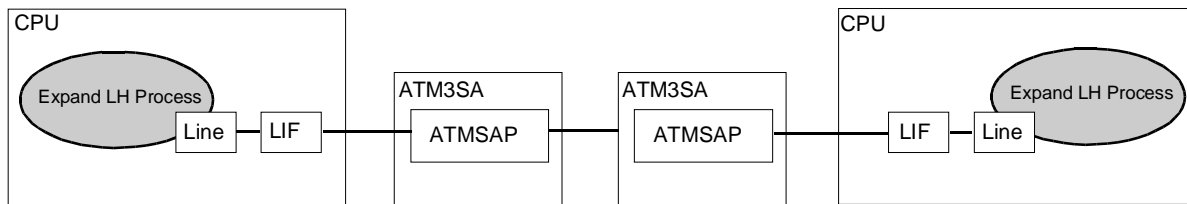
## ATMSAP and Expand

The Expand product connects NonStop NS-series or NonStop S-series systems in networks of up to 256 nodes.

The SLSA ATMSAP offers an ATM Native Mode network interconnect support similar to that offered by the PVC object within an ATM subsystem. Expand issues native mode frames directly to the ATM product via a LIF associated with an ATMSAP object.

[Figure 2-7](#) illustrates ATMSAP use by Expand.

**Figure 2-7. Expand and ATMSAP**



VST016.vsd

## Logical Interface (LIF)

The LIF is a naming point and a logical abstraction of the interface. LAN service providers (see page [2-9](#)) connect to SLSA at the LIF. Clients of the SLSA subsystem refer to a LIF by an object name of the form `$ZZLAN.lif-name`, for example, `$ZZLAN.L018`. (See [HP Manufacturing Naming Conventions \(G06.26 and Earlier RVUs\)](#) on page 3-2 for information about preconfigured naming conventions and suggestions for naming your LIFs.)

Each LIF connects to one PIF.

## Physical Interface (PIF)

The PIF represents the physical connection to the LAN. PIF names have the form `$ZZLAN.adapter-name.sac-unit#.pif-unit-letter`, for example: `$ZZLAN.G4SA1.0.A`. (See [HP Manufacturing Naming Conventions \(G06.26 and Earlier RVUs\)](#) on page 3-2 for information about preconfigured naming conventions and suggestions for naming your PIFs.)

## ServerNet Addressable Controller (SAC)

A SAC is the ServerNet addressable entity on an adapter and can support one or more PIFs. SAC names have the form `$ZZLAN.adaptername.sac-unit#`, for example: `$ZZLAN.E4SA0.0`. (See [HP Manufacturing Naming Conventions \(G06.26 and Earlier RVUs\)](#) on page 3-2 for information about preconfigured naming conventions and suggestions for naming your SACs.)

# Fault Tolerance of the SLSA Subsystem

The SLSA subsystem operates even after a loss of one of its components. This section describes how the SLSA subsystem maintains data paths to the LAN subsystem when a processor, SAC, media, or adapter is unavailable.

## Loss of a ServerNet Fabric

When access through a ServerNet fabric is lost, all traffic goes through the remaining fabric. When the failing fabric is restored, all traffic goes over both fabrics again. If both fabrics fail, traffic moves as described in [Loss of Access of One Processor to a SAC](#) on page 2-16 or [Loss of Access of All Processors to a SAC](#) on page 2-17.

When access through a ServerNet fabric is lost, SLSA events are generated and stored in the Event Management Service (EMS) log. Transient-fault event messages that show a transient-fault number of 3044 (Path switched to X fabric) or 3045 (Path switched to Y fabric) may indicate that access across a fabric was lost.

## Loss of Access of One Processor to a SAC

If a SAC on an adapter becomes unavailable to a LANMON that has a data path to the SAC, all clients (such as TCP/IP, PAM, and IPX/SPX) on that processor lose access, and the SLSA subsystem performs the following steps:

1. The LANMON detects that it cannot communicate with the SAC because:
  - a. The SAC issued a command that timed out.
  - b. The SAC received an error.
  - c. The SAC did not respond after 3 consecutive pings sent 10 seconds apart.
2. The LANMON tries to recover access to the SAC by trying the alternate ServerNet fabric. If the LANMON does not succeed:
  - a. The LANMON informs the LANMAN process about lost access to the SAC.
  - b. If the LANMON that lost access was the owning LANMON, the LANMAN process assigns ownership of the SAC to another processor, but only if the LANMON in that processor can access the SAC.
  - c. LANMAN checks whether any other processors have access, and, if none does, proceeds to the behavior described in [Loss of Access of All Processors to a SAC](#) on page 2-17. If other processors have access to the SAC, LANMAN repeats Step 2b (above).

If access is not reestablished, the clients have to wait until the SAC automatically comes up again.

When access to a SAC is lost, SLSA events are generated and stored in the EMS log. The following events may indicate that the SAC has become unavailable to a processor:

Event Number	Cause
4001	SAC transient fault with a fault number of 2011 to 2014, 2018, 2033, 2036, 2040, 2044, 2045, 3011 to 3018, 3025 to 3027, 3036, 3038, 3043, 3046, or 3047.
4007	SAC ownership change
4009	SAC state after ownership change
4102	PIF available

## Loss of a Processor

If a processor becomes unavailable to a SAC, the SLSA subsystem performs the following steps to maintain the data paths to the adapter:

1. For those SACs where the unavailable processor is the owner, the LANMAN process assigns ownership of the SAC to the next available processor in the access list.
2. All LANMON processes that have access to the SAC notify the SLSA clients (PAM, TCP/IP, or IPX/SPX) that one of the processors in the access list has lost access to the SAC.

When ownership of a SAC changes because a processor is unavailable, SLSA events are generated and stored in the EMS log. The following events may indicate that ownership of the SAC has been reassigned because a processor was unavailable:

Event Number	Cause
4603	LANMON unavailable
4007	SAC ownership change
4009	SAC state after ownership change
4102	PIF available

## Loss of Access of All Processors to a SAC

If a SAC becomes unavailable to all processors, the SAC goes from STARTED to STARTING and the controlling LANMON tries to reestablish the connection to the SAC. If the LANMON is successful, the SAC goes to the STARTED (available) state. The system continues trying to start the SAC until successful.

When access to a SAC is lost, SLSA events are generated and stored in the EMS log. The following events may indicate that the SAC has become unavailable to a processor:

Event Number	Cause
4003	SAC unavailable.
4004	SAC state change. (A SAC that remains in the STARTING state for an unreasonable length of time indicates a problem with the SAC.)
4011	SAC dump file created. (Occurs only if the AUTODUMP option is enabled.)

## Loss of an Adapter

If both SACs on an adapter simultaneously become unavailable, the adapter is unavailable to the system. In this case, the SLSA subsystem performs the same steps described under [Loss of Access of All Processors to a SAC](#) on page 2-17 for both SACs.

## Loss of Media

If a TRSA port is disconnected from the network, clients are notified, the PIF is put in the STARTING state, and the following EMS message is generated:

Event	Cause
4013, transient fault number 3023	One or more of the following conditions cause this fault: <ol style="list-style-type: none"> <li>1. The Token Ring cable was pulled out</li> <li>2. The MAU had a fault</li> <li>3. The Token Ring cable had a fault</li> <li>4. The transceiver had a problem</li> </ol>

Clients are notified when the port connection is restored.

For other port types, the PIF remains in the STARTED state, clients are notified of a change in the link pulse, and the following EMS message is generated:

Event	Cause
4013	The status of the link pulse changed.

# SLSA Subsystem Installation and Configuration

<a href="#">SLSA Subsystem Installation</a>	<a href="#">3-1</a>
<a href="#">SLSA Subsystem Configuration</a>	<a href="#">3-2</a>
<a href="#">HP Manufacturing Naming Conventions (G06.26 and Earlier RVUs)</a>	<a href="#">3-2</a>
<a href="#">HP Manufacturing Naming Conventions (For G06.27 and Later G-Series RVUs and HO6.03 and Later H-Series RVUs and J06.03 and Later J-Series RVUs)</a>	<a href="#">3-6</a>
<a href="#">LAN Configuration</a>	<a href="#">3-9</a>

---

**Note.** For instructions about how to install or replace adapters, see [Adapter Manuals](#) on page 1-7.

---

## SLSA Subsystem Installation

The SLSA subsystem is preinstalled and preconfigured on Integrity NonStop NS-series or NonStop S-series servers and starts during the system-load sequence.

During the system-load sequence, the persistence manager (\$ZPM) starts the primary and backup LAN Manager (LANMAN) processes (\$ZZLAN) in processors 0 and 1.

The LANMAN process then starts the LAN Monitor (LANMON) processes in each processor and tries to assign ownership of each ServerNet addressable controller (SAC) to a LANMON process.

Each LANMON process then initializes the adapter, SAC, physical interface (PIF) and logical interface (LIF) objects that it owns, reporting any state transitions to other LANMON processes.

---

**Note.** Subsystem Control Facility (SCF) commands issued to the LANMONs or the LANMAN are not accepted while they are starting or when the backup LANMAN is taking over from the primary LANMAN. The SLSA subsystem sends the error message: "SLSA process is busy...."

---

# SLSA Subsystem Configuration

Your Integrity NonStop NS-series or NonStop S-series server is shipped with a generic preconfiguration of the SLSA subsystem, which you can modify. However, changes made to the configuration might also influence the placement and configuration of the connections to any ServerNet wide area network (SWAN) concentrators you plan to attach to your servers.

---

**Note.** HP recommends that all SLSA objects comply with the following naming specification: eight alphanumeric characters, with the first character being a letter.

---

## HP Manufacturing Naming Conventions (G06.26 and Earlier RVUs)

For G06.26 and earlier RVUs, HP manufacturing uses a naming convention for processes and devices that relates logical names to the physical location of devices.

---

**Table 3-1. Naming Convention for SLSA Subsystem and Related Processes (G06.26 and Earlier RVUs)**

---

Process or Device Type	Convention	Example
LAN Manager (LANMAN) process	Must be \$ZZLAN	\$ZZLAN
LAN Monitor (LANMON) process	<i>\$ZZLAN.#ZLMprocessorid</i>	<i>\$ZZLAN.#ZLM01</i>
Logical interface (LIF)	<i>\$ZZLAN.Lcabid-portid</i>	<i>\$ZZLAN.L018</i>
LAN adapters (E4SA, FESA, GESA, G4SA, CCSA, or TRSA) (See <a href="#">Suggested Naming Conventions for Adapters</a> on page 3-7 for details.)	<i>\$ZZLAN.Ecabid-slot</i> For G4SAs, see <a href="#">G4SA Naming Conventions (G06.24, G06.25, and G06.26 RVUs)</a> on page 3-4.	<i>\$ZZLAN.E0153</i>
Processor multifunction (PMF) or I/O multifunction (IOMF) CRU	<i>\$ZZLAN.MIOEprocessorid</i>	<i>\$ZZLAN.MIOE1</i>
NonStop TCP/IP process	<i>\$ZBcabid-portid</i>	<i>\$ZB018</i>
SWAN or SWAN 2 concentrator	<i>Sconcnun</i>	S01

*processorid*

specifies the processor in which the LANMON process is running. For example, #ZLM01 specifies the LANMON in processor 1.



*cabid*

is the two-digit number that identifies the enclosure.

<b><i>cabid</i></b>	<b>Description</b>	<b>Range of Values</b>
0 <i>n</i>	The object is in processor enclosure <i>n</i> .	01-08
<i>n</i> 1	The object is in I/O enclosure 1 attached to processor enclosure <i>n</i> .	11, 21, 31, 41, 51, 61, 71, or 81
<i>n</i> 2	The object is in I/O enclosure 2 attached to processor enclosure <i>n</i> .	12, 22, 32, 42, 52, 62, 72, or 82
<i>n</i> 3	The object is in I/O enclosure 3 attached to processor enclosure <i>n</i> .	13, 23, 33, 43, 53, 63, 73, or 83

*portid*

is the combination of the slot number and port number mapped in the following way:

<b>Slot Number</b>	<b>Port Number</b>	<b><i>portid</i></b>	<b>Slot Number</b>	<b>Port Number</b>	<b><i>portid</i></b>
51	0	0	53	0	8
51	1	1	53	1	9
51	2	2	53	2	A
51	3	3	53	3	B
52	0	4	54	0	C
52	1	5	54	1	D
52	2	6	54	2	E
52	3	7	54	3	F

*slot*

is the actual physical slot number in the enclosure:

<b>Number</b>	<b>Description</b>
50 or 55	PMF or IOMF CRU
51 to 54	A LAN adapter

## G4SA Naming Conventions (G06.24, G06.25, and G06.26 RVUs)

Process or Device Type	Convention	Example
LIF object	<i>Lcabid-module-portid</i>	\$ZZLAN.L112A
G4SA ADAPTER object	<i>Ggroup-module-slot</i> A G4SA must be installed in an I/O adapter module (IOAM) enclosure.	\$ZZLAN.G1123
Processor multifunction (PMF) or I/O multifunction (IOMF) CRU	<i>\$ZZLAN.MIOEprocessorid</i>	\$ZZLAN.MIOE1
TCP/IP process	<p><i>\$ZBprocessor cabid-IOAM position 1-5 + module-portid</i></p> <p><b>Considerations:</b></p> <ul style="list-style-type: none"> <li>● \$ZB is reserved in the HP NonStop operating system for the NonStop TCP/IP process</li> <li>● Every IOAM group contains two modules. Twenty G4SA ports are available for each module, and forty G4SA ports are available for each IOAM group. The TCP/IP naming convention is limited to six characters: the three initial characters are reserved (\$ZB) and the final three characters must include: <i>processor cabid-IOAM position 1-5 + module-portid</i></li> <li>● <i>portid</i> can be any letter from A to U; these letters can be repeated for modules 2 and 3.</li> </ul>	\$ZB1AA
TELSERV process	\$ZN (plus the last three characters of the NonStop TCPIP process name)	\$ZN1AA
LISTNER process	\$ZP (plus the last three characters of the NonStop TCPIP process name)	\$ZP1AA

*cabid*

is the two-digit group number that identifies the processor enclosure that is connected to the IOAM or to the IOAM enclosure that contains the G4SA; cabid is also the group number.

<b>group</b>	<b>Description</b>	<b>Range of Values</b>
0 <i>n</i>	The object is in processor enclosure <i>n</i> .	01-08
1 <i>n</i>	The object is in I/O enclosure <i>n</i> . The IO enclosure can be physically attached to as many as four processors in the inner tetrahedron, but will always have the processor enclosure number 1.	11,12,13,14,15

*module*

identifies whether the G4SA is installed in module 2 or 3 of an IOAM. A module is either 2 or 3.

*slot*

is the G4SA's physical slot number in the IOAM:

<b>Number</b>	<b>Description</b>
1 to 5	For the G4SA.
The supported G4SA installation slots differ by system type. To determine the supported installation slots for your G4SA, see the planning guide for your system (for example, the <i>Integrity NonStop NS1000 Planning Guide</i> )	

*processorid*

specifies the processor in which the LANMON process is running. For example, #ZLM01 specifies the LANMON in processor 1.

*IOAM position 1-5 + Module*

is the IOAM group's position in the topology where one of the IOAMs is connected to a processor enclosure. This convention uses alphabetical letters A through J:

<b>A</b>	<b>B</b>	<b>C</b>	<b>D</b>	<b>E</b>	<b>F</b>	<b>G</b>	<b>H</b>	<b>I</b>	<b>J</b>
1, 2	1, 3	2, 2	2, 3	3, 2	3, 3	4, 2	4,3	5, 2	5, 3

In the above table, A is Module 2 of the IOAM connected to the first port of the processor enclosure's ServerNet Expansion Board (SEB).

*portid*

is the combination of the slot number and port number mapped in the following way for G4SA; this convention uses alphabetical letters A through U:

Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Port Number
A	E	I	M	R	A
B	F	J	N	S	B
C	G	K	P	T	C
D	H	L	Q	U	D

In the above table, G is the portid for Port C of the G4SA installed in slot 2 of an IOAM.

## HP Manufacturing Naming Conventions (For G06.27 and Later G-Series RVUs and H06.03 and Later H-Series RVUs and J06.03 and Later J-Series RVUs)

For G06.27 and later G-series RVUs, and H06.03 and later H-series RVUs and J06.03 and later J-series RVUs, HP manufacturing uses a naming convention for processes and devices that relates logical names to the physical location of devices. For more detailed J-series naming conventions, refer to the applicable planning guide (for example, the *NonStop BladeSystem Planning Guide*).

**Table 3-2. Naming Convention for SLSA Subsystem and Related Processes (G06.27 and Later G-Series RVUs, and H06.03 and Later H-Series RVUs, and J06.03 and Later J-Series RVUs)**

Process or Device Type	Convention	Example
LAN Manager (LANMAN) process	Must be \$ZZLAN	\$ZZLAN
LAN Monitor (LANMON) process	\$ZZLAN.#ZLM <i>processorid</i>	\$ZZLAN.#ZLM01
G4SA adapter object	G <i>group-module-slot</i>	\$ZZLAN.G11123
G4SA LIF object	L <i>group-module-slot-PIF</i>	\$ZZLAN.L11123A
TCP/IP process	\$ZTC <i>number</i> is the first TCPSAM process for the system	\$ZTC0
SWAN concentrator	S <i>concnm</i>	S01

*processorid*

specifies the process in which the LANMON is running. For example, #ZLM01 specifies the LANMON in processor 1.

*group*

is in the range of 110 through 115 for IOAM enclosures installed in NonStop NS16000 systems or NonStop BladeSystems. For IOAM enclosures installed in NonStop NS14000 and NS1000 servers and for VIO enclosures, the group is 100. (VIO enclosures are available in H06.08 and subsequent H-series RVUs.)

*module*

identifies whether the G4SA is installed in module 2 or 3 of an IOAM or VIO enclosure. A module is either 2 or 3.

*slot*

is the G4SA's physical slot number in the IOAM or the slot numbers in a VIO enclosure that contains the Ethernet ports.

---

**Note.** Supported G4SA installation slots differ by system model. To determine the supported installation slots for your adapters, see the planning guide for your system (for example, the *NonStop NS1000 Planning Guide*). Ethernet ports in the VIO enclosure, also represented by the G4SA adapter type in SLSA, go in slots 6 and 7. For information about the VIO enclosure, see the *Versatile I/O Manual*.

---

*port*

is the G4SA or VIO enclosure Ethernet port physical interface (PIF). A PIF is part of the SLSA subsystem and represents the physical port on the G4SA or the physical port in slot 6 or 7 of the VIO enclosure.

## Suggested Naming Conventions for Adapters

Here are some guidelines for naming adapters:

- Follow the SLSA naming specification: no more than eight alphanumeric characters, with the first character being a letter
- Base that first letter character on the type of adapters in your system. For example:
  - Use an A for ATM (for example A0153), C for CCSA, E for E4SA, F for FESA, G for GESA, G for G4SA, or T for TRSA. Use the SCF [ADD ADAPTER Command](#) on page 4-15 to name your adapters.
  - A name that started with the letter M would indicate the Ethernet port on the processor multifunction CRU.
- Identify the adapter by using a number based on the processor enclosure, IOAM enclosure, or I/O enclosure in which the adapter is installed.

- Add the slot number and module number that the adapter is in to the adapter name. For example, a G4SA that is in slot 3 of module 2 of IOAM Group 111, would have the name G11123.

## Suggested Naming Conventions for ATMSAP Object

`$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id`

*adapter-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that uniquely identifies the target SLSA ADAPTER object. Wildcard characters are allowed.

*sac-id*

is a number that identifies the target SAC object. The only valid *sac-id* for the ATM3SA adapter is the number “0.” Wildcard characters are allowed.

*pif-id*

is an alpha character that identifies the target PIF object. The only valid *pif-id* for the ATM3SA adapter is the letter “A.” Wildcard characters are allowed.

*atmsap-id*

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP object. Wildcard characters are allowed.

An example of an ATMSAP object name is:

`$ZZLAN.ATM01.0.A.ATMSAP01`

## Suggested Naming Conventions for LIFs

A LIF name begins with the letter L followed by the adapter’s physical location (for example, group, module, slot) and the port identifier to indicate the physical port on the adapter or in slot 6 or 7 of the VIO enclosure with which the LIF corresponds.

Depending on your RVU and adapter type, use one of these port identifier methods.

- For NonStop systems not accessing G4SAs or running G06.23 and earlier RVUs, see [portid](#) on page 3-3.
- For NonStop systems and IOAM enclosures using G4SAs and running G06.24 through G06.26 RVUs, see [portid](#) on page 3-6.
- For NonStop systems and IOAM enclosures using G4SAs and running G06.27 and later RVUs or H06.03 and later RVUs or J06.03 and later RVUs, see [port](#) on page 3-7.
- For the VIO enclosure, see [port](#) on page 3-7.

## Suggested Naming Conventions for SACs and PIFs

The SLSA subsystem automatically names the SACs and PIFs based on the name you assign to the adapters. For example, if you assign the name G11123 to a G4SA, the system names the SAC G11123.0. The PIFs for SAC G11123.0 are G11123.0.A, G11123.0.B, G11123.0.C, and G11123.0.D.

## LAN Configuration

1. Complete configuration forms for each PMF CRU (NonStop S-series servers only) and adapter in the system. Write a command file to configure the adapters or modify the configuration file shipped with your system:
  - a. See [Figure 3-1](#) for a sample PMF CRU form. A blank copy of this form is also at the end of this section.
  - b. See [Adapter Manuals](#) on page 1-7 to determine which manual has your configuration form. [Figure 3-2](#) shows a sample completed G4SA Configuration Form.
2. When you have finished the configuration forms, place them in your Documentation Packet. For more information about completing a PMF CRU configuration form, see the *NonStop S-Series Planning and Configuration Guide*. For more information about the Documentation Packet, see the planning guide for your server model (for example, the *NonStop NS-Series Planning Guide*).

**Figure 3-1. Example of a Completed Processor Multifunction (PMF) CRU Configuration Form (For NonStop S-Series Servers Only)**

Processor Multifunction (PMF) CRU Configuration Form

System Name \Case1

Date 01 / 21 / 00

Shaded areas indicate nonconfigurable components

Group 01Module 01Slot 50

SCSI Port

Product Number: 5794

SCF Name: \$TAPE0

SCSI Cable: PN 131369

Ethernet Port

IP Address: Initially 192.231.36.10  
Get new address from LAN department

Adapter Name: \$ZZLAN.MIOE0

SAC Name: \$ZZLAN.MIOE0.0

SAC Access List: 0,1

PIF Name: \$ZZLAN.MIOE0.0.A

LIF Name: \$ZZLAN.LANX

POWER ON

SCSI

SERIAL CONSOLE

ETHERNET

MODEM

AC Power (S7000)

DC Power (S70000)

AU X

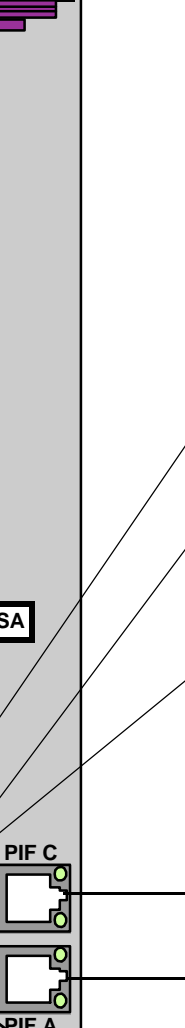
POWER-ON CABLE

VST207.vsd

LAN Configuration and Management Manual—520469-011  
3-10



**Figure 3-2. Example of a Completed Gigabit Ethernet 4-Port ServerNet Adapter (G4SA) Configuration Form**



System Name Case 1

Date 11/04/04

Group 111    Module 2    Slot 3

Adapter Name: G11123 IP Address: 192.231.036.100

SAC Name: G11123.0 SAC Access List: (0, 1, 2, 3)

PIF Name: G11123.0.D LIF Name: L11123D

Adapter Name: G11123 IP Address: 192.231.036.200

SAC Name: G11123.0 SAC Access List: (0, 1, 2, 3)

PIF Name: G11123.0.C LIF Name: L11123C

Adapter Name: G11123 IP Address: 192.231.036.300

SAC Name: G11123.0 SAC Access List: (0, 1, 2, 3)

PIF Name: G11123.0.D LIF Name: L11123D

Adapter Name: G11123 IP Address: 192.231.036.400

SAC Name: G11123.0 SAC Access List: (0, 1, 2, 3)

PIF Name: G11123.0.C LIF Name: L11123C

Adapter Name: G11123 IP Address: 192.231.036.500

SAC Name: G11123.0 SAC Access List: (0, 1, 2, 3)

PIF Name: G11123.0.A LIF Name: L11123A

Adapter Name: G11123 IP Address: 192.231.036.600

SAC Name: G11123.0 SAC Access List: (0, 1, 2, 3)

PIF Name: G11123.0.B LIF Name: L11123B

**Only four of the G4SA's six ports can be in use at any one time.**

VST003.vsd

## Configuration File

To configure the adapters, use a command file to:

- Add the adapters and LIFs to the system
- Start the adapter and its subordinate objects
- Associate LIFs with each PIF
- Start the LIFs
- Assign an IP address to each LIF

The following TACL command file configures a single G4SA adapter:

```
?TACL MACRO

PUSH #INLINEPREFIX
SET VARIABLE #INLINEPREFIX +
SCF /INLINE, OUT [#MYTERM], NAME/
+ ASSUME PROCESS $ZZLAN
+ ADD ADAPTER G11123, TYPE G4SA, LOCATION (111,2,3), ACCESSLIST (0,1,2,3)
+ START ADAPTER G11123, SUB ALL
+ ADD LIF L11123A, PIF G11123.0.A
+ ADD LIF L11123B, PIF G11123.0.B
+ ADD LIF L11123C, PIF G11123.0.C
+ ADD LIF L11123D, PIF G11123.0.D
+ START LIF *
+ EXIT
POP #INLINEPREFIX
```

The following TACL command file configures the Ethernet ports in a VIO enclosure (available in H06.08 and subsequent H-series RVUs):

```
?TACL MACRO

PUSH #INLINEPREFIX
SET VARIABLE #INLINEPREFIX +
SCF /INLINE, OUT [#MYTERM], NAME/
+ ASSUME PROCESS $ZZLAN
+ ADD ADAPTER G10026, TYPE G4SA, LOCATION (100,2,6), ACCESSLIST (0,1,2,3)
+ START ADAPTER G10026, SUB ALL
+ ADD LIF L10026A, PIF G10026.0.A
+ ADD LIF L10026B, PIF G10026.0.B
+ ADD LIF L10026C, PIF G10026.0.C
+ ADD LIF L10026D, PIF G10026.0.D
+ START LIF *
+ EXIT
POP #INLINEPREFIX
```

# SCF Commands for the SLSA Subsystem

This section describes the Subsystem Control Facility (SCF) commands interpreted for the SLSA subsystem:

<a href="#">Supported Commands and Object Types</a>	Page <a href="#">4-2</a>
<a href="#">SCF Objects</a>	Page <a href="#">4-4</a>
<a href="#">Generic Processes</a>	Page <a href="#">4-6</a>
<a href="#">ABORT Command</a>	Page <a href="#">4-7</a>
<a href="#">ADD Command</a>	Page <a href="#">4-15</a>
<a href="#">ALTER Command</a>	Page <a href="#">4-27</a>
<a href="#">DELETE Command</a>	Page <a href="#">4-39</a>
<a href="#">INFO Command</a>	Page <a href="#">4-41</a>
<a href="#">LISTOPENS Command</a>	Page <a href="#">4-63</a>
<a href="#">NAMES Command</a>	Page <a href="#">4-64</a>
<a href="#">RESET Command</a>	Page <a href="#">4-75</a>
<a href="#">START Command</a>	Page <a href="#">4-76</a>
<a href="#">STATISTICS Command</a>	Page <a href="#">4-83</a>
<a href="#">STATUS Command</a>	Page <a href="#">4-105</a>
<a href="#">STOP Command</a>	Page <a href="#">4-128</a>
<a href="#">SWITCH Command</a>	Page <a href="#">4-134</a>
<a href="#">TRACE Command</a>	Page <a href="#">4-135</a>
<a href="#">VERSION Command</a>	Page <a href="#">4-149</a>

# Supported Commands and Object Types

This section describes the Subsystem Control Facility (SCF) commands you can use to manage the SLSA subsystem. [Table 4-1](#) lists the SLSA SCF commands and objects along with the page numbers where those commands are described in this manual. (A page number in the object column indicates that the command is supported on that object.) For a detailed description of SCF, refer to the *SCF Reference Manual for G-Series RVUs* or *SCF Reference Manual for J-Series and H-Series RVUs*.

**Table 4-1. SCF Commands and Objects** (page 1 of 2)

Command	Objects						
	ADAPTER	LIF	MON	PIF	PROCESS	SAC	ATMSAP
<a href="#">ABORT Command</a> on page 4-7	<a href="#">4-7</a>	<a href="#">4-10</a>	<a href="#">4-11</a>	<a href="#">4-12</a>		<a href="#">4-13</a>	<a href="#">4-9</a>
<a href="#">ADD Command</a> on page 4-15	<a href="#">4-15</a>	<a href="#">4-23</a>					<a href="#">4-21</a>
<a href="#">ALTER Command</a> on page 4-27		<a href="#">4-29</a>		<a href="#">4-32</a>		<a href="#">4-36</a>	<a href="#">4-27</a>
<a href="#">DELETE Command</a> on page 4-39	<a href="#">4-39</a>	<a href="#">4-40</a>					<a href="#">4-39</a>
<a href="#">INFO Command</a> on page 4-41	<a href="#">4-41</a>	<a href="#">4-46</a>	<a href="#">4-49</a>	<a href="#">4-50</a>		<a href="#">4-60</a>	<a href="#">4-44</a>
<a href="#">LISTOPENS Command</a> on page 4-63		<a href="#">4-63</a>					
<a href="#">NAMES Command</a> on page 4-64	<a href="#">4-67</a>	<a href="#">4-69</a>	<a href="#">4-71</a>	<a href="#">4-71</a>	<a href="#">4-72</a>	<a href="#">4-75</a>	<a href="#">4-67</a>
<a href="#">RESET Command</a> on page 4-75	<a href="#">4-76</a>						
<a href="#">START Command</a> on page 4-76	<a href="#">4-76</a>	<a href="#">4-79</a>	<a href="#">4-79</a>	<a href="#">4-80</a>		<a href="#">4-81</a>	<a href="#">4-77</a>
<a href="#">STATISTICS Command</a> on page 4-83				<a href="#">4-84</a>			<a href="#">4-83</a>

**Table 4-1. SCF Commands and Objects** (page 2 of 2)

Command	Objects						
	ADAPTER	LIF	MON	PIF	PROCESS	SAC	ATMSAP
<a href="#">STATUS Command</a> on page 4-105	<a href="#">4-106</a>	<a href="#">4-109</a>	<a href="#">4-111</a>	<a href="#">4-113</a>	<a href="#">4-123</a>	<a href="#">4-125</a>	<a href="#">4-107</a>
<a href="#">STOP Command</a> on page 4-128	<a href="#">4-128</a>	<a href="#">4-130</a>		<a href="#">4-131</a>		<a href="#">4-133</a>	<a href="#">4-129</a>
<a href="#">SWITCH Command</a> on page 4-134					<a href="#">4-134</a>		
<a href="#">TRACE Command</a> on page 4-135			<a href="#">4-135</a>	<a href="#">4-138</a>	<a href="#">4-142</a>	<a href="#">4-145</a>	
<a href="#">VERSION Command</a> on page 4-149			<a href="#">4-149</a>		<a href="#">4-151</a>		

## Command Cancellation

The SLSA subsystem does not support cancellation of SCF commands. Once you have issued an SCF command, you cannot halt an initiated operation.

## Sensitive and Nonsensitive Commands

Some SCF commands are sensitive and special qualification is required to use them. Sensitive commands can change the state or configuration of subsystem objects, start or stop tracing, or change the values of statistics counters. The use of sensitive commands is limited to the following user IDs:

- Members of the super ID (group ID 255)
- Members of the user group that owns the process to which the command is sent

Other SCF commands are nonsensitive, meaning that they request information or status but do not affect operation; these commands are available to all users. [Table 4-2](#) lists the sensitive and nonsensitive SCF commands for the SLSA subsystem.

**Table 4-2. Sensitive and Nonsensitive SCF Commands**

<b>Sensitive Commands</b>	<b>Nonsensitive Commands</b>
ABORT	INFO
ADD	LISTOPENS
ALTER	NAMES
DELETE	STATS (without RESET option)
RESET	STATUS
START	VERSION
STOP	
STATS (with RESET option)	
SWITCH	
TRACE	

## SCF Objects

SLSA supports the following SCF objects:

- ADAPTER
- ATMSAP
- LIF
- PIF
- PROCESS
- SAC

The ADAPTER object represents the customer-replaceable unit (CRU) and provides the SCF interface for managing the adapters.

The ATMSAP object represents an ATM-protocol-direct service access point interface. It provides direct access to a VCC for Expand.

The LIF object represents the logical interface on the adapter and provides the SCF interface for managing the LIFs. LIF objects are associated with either a PIF object or an ATMSAP object.

The PIF object represents the physical interface and provides the SCF interface for managing the PIFs.

The PROCESS object is used to control different aspects of the SLSA subsystem. The manager process is called LANMAN and controls all LAN connectivity including receiving DSM requests, starting the LAN Monitor (LANMON) process in each processor and maintaining them persistently across failures, and assigning SACs to LANMON processes. LANMON controls access to adapters, reporting driver interrupt handler (DIH) errors, provides tracing, broadcasts state changes for SACs and receives Simple Network Management Protocol (SNMP) requests.

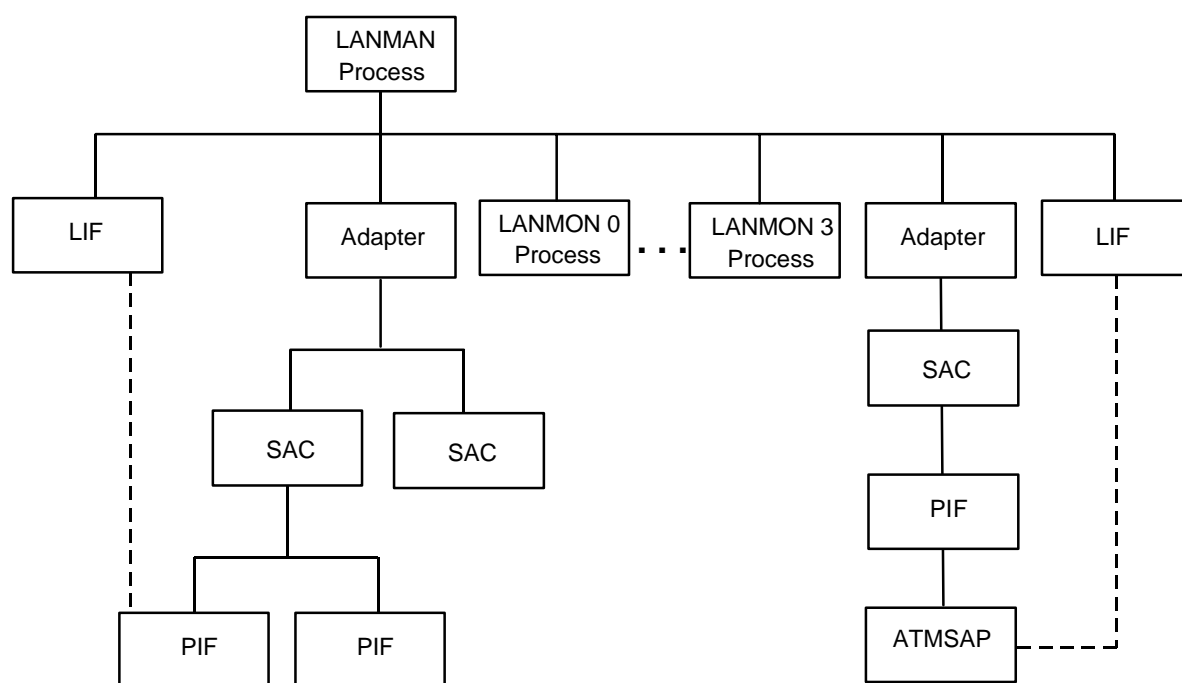
The SAC object represents the ServerNet addressable controller and provides the SCF interface for managing the SACs.

## SCF Object Hierarchy

The SCF objects for the SLSA subsystem correspond to process and hardware components within the subsystem. Some objects are subordinate to others. [Figure 4-1](#) shows the hierarchy of the objects in the SLSA subsystem.

**Note.** The PIF object is not subordinate to the LIF. The LIF has a logical relationship to a PIF object indicated by the dotted line in [Figure 4-1](#).

**Figure 4-1. SCF Object Hierarchy**



VST802.vsd

# SCF Object States

SCF objects can be in different states of operation, depending on the SCF command issued to that object. The operational states are:

- STARTED

The object is running and is ready to accept requests from other subsystem components. This state results from the START command for most objects.
- STARTING

The object is on its way to the STARTED state because a command was issued to start it.
- STOPPING

The object is on its way to the STOPPED state because a command was issued to stop it.
- STOPPED

The object does not accept requests from other subsystem components. This state results from the STOP or ABORT command.
- DIAGNOSING

A diagnostic process is testing the object.

[Table 4-3](#) lists the operational states for the objects in the SLSA subsystem.

**Table 4-3. Operational States for SLSA Objects**

Object	Possible States				
	STOPPING	STOPPED	STARTING	STARTED	DIAGNOSING
ADAPTER		X	X	X	
LIF		X		X	
PIF		X	X	X	
SAC		X	X	X	X
MONITOR	X	X	X	X	
PROCESS				X	
ATMSAP		X	X	X	

## null Object Type

The null object type is not an actual object type. The term null represents the lack of a specified object. Any SCF command that supports the null object type is issued without the specification of an object type. Commands support the null object type if an object type is irrelevant or if they refer to a collection of objects.

# Generic Processes

The operating system or a user creates a generic process to perform a task. The LAN Manager (LANMAN) process is a generic process created by the HP NonStop Kernel persistence monitor and configured as a generic process. The default user ID for a generic process is the same as the user ID for that SCF session.



Generic processes have the following characteristics:

- You can configure a generic process to start in one processor, in more than one processor, or in each processor in the system.
- You can permanently change an attribute of a generic process by using the SCF ALTER command. This change takes effect the next time the process is started.
- You can create a generic process to run at a high pin or a low pin. The default is to run at a high pin.
- You cannot use TACL ASSIGNS, DEFINEs, or PARAMs on a generic process. Only the SCF ADD and ALTER commands can create or change the behavior of a generic process.
- You can configure a generic processes as persistent (to restart automatically if stopped abnormally).

Refer to the *SCF Reference Manual for the Kernel Subsystem* for more detailed information about generic processes.

## ABORT Command

ABORT is a sensitive command used to halt an object's operation as quickly as possible. Only enough processing is done to ensure that the object can run again when it is restarted.

### ABORT ADAPTER Command

The ABORT ADAPTER command halts the operation of the specified ADAPTER as quickly as possible. Only enough processing is done to ensure that the object can run again when restarted. The object is left in the STOPPED state if the command is successful.

### Command Syntax

```
ABORT [ /OUT file-spec/ ] ADAPTER adapter-name
      [ , SUB [ ONLY | ALL | NONE ] ]
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*adapter-name*

is the name of the ADAPTER to be aborted. The ADAPTER name has the form \$ZZLAN.*adapter-name*, for example \$ZZLAN.G11123.

SUB [ ONLY | ALL | NONE ]

directs the command at a set of subordinate objects.

ONLY specifies that only the subordinate objects are affected.

ALL specifies that the named object and the subordinate objects are affected.

NONE specifies that none of the subordinate objects are affected.

## Considerations

- The ABORT ADAPTER command does not work if subordinate ServerNet addressable controllers (SACs), logical interfaces (LIFs), physical interfaces (PIFs), and ATMSAPs are not in the STOPPED state, unless you specify the SUB ALL option.
- To restart an aborted adapter, use the START command.
- Use the STATUS command to determine the current summary state of the adapter.
- If a subordinate SAC is in the DIAGNOSING state, you cannot abort the adapter.
- Omitting the SUB option has the same effect as SUB NONE.
- Including the SUB option without specifying ALL, NONE, or ONLY has the same affect as SUB ALL.

## Examples

The following are examples of the ABORT ADAPTER command:

```
ABORT ADAPTER $ZZLAN.G11123
```

```
ABORT ADAPTER ($ZZLAN.G11123, $ZZLAN.G11235)
```

```
ABORT ADAPTER $ZZLAN.G*
```

```
ABORT ADAPTER $ZZLAN.G11235, SUB ALL
```

# ABORT ATMSAP Command

The ABORT ATMSAP command forces an ATMSAP object subordinate to a PIF object to halt operation as soon as possible. The object enters the STOPPED summary state if the command is successful. The command is rejected if the ATMSAP is already in the STOPPED summary state.

## Command Syntax

```
ABORT [ /OUT file-spec/ ] ATMSAP atmsap-name
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*atmsap-name*

is the name of the target ATMSAP objects. *atmsap-name* is in the form

*\$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id*

where:

*adapter-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that uniquely identifies the target SLSA ADAPTER objects. Wildcard characters are allowed.

*sac-id*

is a number that identifies the target SAC objects. The only valid *sac-id* for the ATM3SA adapter is the number "0." Wildcard characters are allowed.

*pif-id*

is an alpha character that identifies the target PIF objects. The only valid *pif-id* for the ATM3SA adapter is the letter "A." Wildcard characters are allowed.

*atmsap-id*

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP objects. Wildcard characters are allowed.

## Response Display

The ABORT ATMSAP command returns only a success or failure indication. Successful completion is indicated when SCF displays the prompt for the next command. Failure is indicated when SCF displays an error message.

## Example

```
ABORT ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01
```

## ABORT LIF Command

The ABORT LIF command stops the operation of the specified logical interface (LIF) as quickly as possible and places the LIF in the STOPPED state. Only enough processing is done to ensure that the LIF can run again when it is restarted.

## Command Syntax

<pre>ABORT [ /OUT <i>file-spec</i>/ ] LIF <i>lif-name</i></pre>
---

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*LIF lif-name*

is the name of the LIF to be aborted. The LIF name has the form *\$ZZLAN.lif-name*, for example, *\$ZZLAN.L11123B*.

## Considerations

- Use the ABORT LIF command when you need to stop the operation of a LIF immediately and the STOP command does not work, is unavailable, or clients are using the LIF.
- All activities that the LIF is performing are terminated when the LIF is aborted, which may cause clients of the SLSA subsystem that are using the specified LIF to be left in an inconsistent or incomplete state.
- The ABORT LIF command has no effect on the state of the associated PIF.

## Examples

The following are examples of the ABORT LIF command:

```
ABORT LIF $ZZLAN.L11123B
```

```
ABORT LIF ($ZZLAN.L11123A, $ZZLAN.L11123B)
```

```
ABORT LIF $ZZLAN.L*
```

## ABORT MON Command

The ABORT MON command stops the operation of the specified LAN Monitor (LANMON) and places the LANMON in the STOPPED state.

### Command Syntax

```
ABORT [ /OUT file-spec/ ] MON lanmon
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

MON *lanmon*

is the name of the LANMON to be aborted. The LANMON name has the form \$ZZLAN.*lanmon*, for example, \$ZZLAN.#ZLM01.

### Considerations

- The ABORT MON command will not work on a LANMON that is running in the same CPU as the primary LANMAN.
- No wildcards are allowed with this command.
- 

#### Effects of the ABORT MON command on Clients/NonStop TCP/IP (page 1 of 2)

**SLSA Clients** When a LANMON is ABORTED, any SLSA clients lose LIF/PIF access from the associated CPU. LANMON delays five seconds after receiving the ABORT message after which LANMON removes its QIO queues and detaches from the QIO segment. At this point any clients that call `SM_LIF_GET_INFO_()`, from the CPU in which no LANMON is running, get error `zsm_err_dr_inv_dev_name`. The delay in LANMON allows clients enough time to call `SM_LIF_GET_INFO_()` in case they need to switch

**Effects of the ABORT MON command on Clients/NonStop TCP/IP** (page 2 of 2)

Other Clients	When a LANMON is aborted, other SLSA clients should handle the situation like NonStop TCP/IP.  Other clients that may be affected include PAM, SS7/SHTI, and IPX/SPX. TSM/OSM and SNMP clients should not be affected.
SS7/SHTI	For SHTI clients running in the same CPU in which a LANMON has been aborted, the calls to <i>slsa_connect_nw</i> and <i>slsa_lif_get_info</i> fail with error <i>slsa_err_invalid_device_name</i> (110). Like other clients, the SHTI client gets an indication that LIF access has been lost whenever a LANMON is aborted
NonStop TCP/IP	When a LANMON is ABORTED, NonStop TCP/IP (NonStop TCP/IP, Parallel Library TCP/IP, and NonStop TCP/IPv6) gets an indication that the PIF/LIF has lost access from its primary CPU and that <i>SM_LIF_GET_INFO</i> has been called. TCP/IP switches and keeps the SUBNET started. LANMON cleans up and stops. TCP/IP fault-tolerant capability is restored once the aborted LANMON is re-started.

## Examples

The following example aborts the LANMON in processor 01:

```
ABORT MON $ZZLAN.#ZLM01
```

The following example aborts the LANMONs in processors 01, 02, and 03:

```
ABORT MON ($ZZLAN.#ZLM01, $ZZLAN.#ZLM02, $ZZLAN.#ZLM03)
```

## ABORT PIF Command

The ABORT PIF command stops the operation of the specified PIF as quickly as possible and places the PIF in the STOPPED state. Only enough processing is done to ensure that the PIF can run again when the PIF is restarted.

This command takes effect immediately.

## Command Syntax

```
ABORT [ /OUT file-spec/ ] PIF pif-name
      [ , SUB { ONLY | ALL | NONE } ]
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PIF *pif-name*

is the name of the PIF to be aborted. The PIF name has the form *\$ZZLAN.adapter-name.sac-unit#.pif-unit-letter*, for example, *\$ZZLAN.G11123.0.A*.

SUB { ONLY | ALL | NONE }

controls the set of objects and subordinate objects that the command targets:

- ONLY specifies that only subordinate objects are targets of the command.
- ALL specifies that the named object and the subordinate objects are targets of the command. This is the default used if the SUB keyword is used but no option is selected.
- NONE specifies that none of the subordinate objects are targets of the command. This is the default selected if the SUB keyword is not used.

## Considerations

- Use the ABORT PIF command when you need to stop the operation of a PIF immediately and the STOP command does not work, or is unavailable, or when clients are using the PIF.
- All activities that the PIF is performing are terminated when the PIF is aborted, which may cause clients of the SLSA subsystem that are using the specified PIF to be left in an inconsistent or incomplete state.
- To restart an aborted PIF, use the START PIF command.
- Use the STATUS PIF command to determine the current summary state of the PIF.
- Aborting a PIF causes the access state of the associated LIF to be DOWN.

## Example

The following example aborts the PIF named G11123.0.A:

```
ABORT PIF $ZZLAN.G11123.0.A
```

The following example aborts a PIF on an E4SA, an MFIOB, a FESA, and a TRSA:

```
ABORT PIF ($ZZLAN.E0153.1.A, $ZZLAN.M0IE0.0.A,&
$ZZLAN.F0152.0.A, $ZZLAN.T0154.0.A)
```

## ABORT SAC Command

The ABORT SAC command stops the operation of the specified SAC as quickly as possible and places the SAC in the STOPPED state. Only enough processing is done to ensure that the SAC can run again when it is restarted.

This command takes effect immediately.

## Command Syntax

```
ABORT [ /OUT file-spec/ ] SAC sac-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*SAC sac-name*

is the name of the SAC to be aborted. The SAC name has the form *\$ZZLAN.adapter-name.sac-unit#*, for example, *\$ZZLAN.G11123.0*.

*SUB [ ONLY | ALL | NONE ]*

directs the command at a set of subordinate objects.

ONLY specifies that only the subordinate objects are affected.

ALL specifies that the named object and the subordinate objects are affected.

NONE specifies that none of the subordinate objects are affected.

## Considerations

- Use the ABORT SAC command when you need to halt immediately the operation of a SAC and the STOP command either does not work or is unavailable.
- Omitting the SUB option has the same affect as SUB NONE.
- Including the SUB option without specifying ALL, NONE, or ONLY has the same affect as SUB ALL.
- Aborting the SAC causes the access state of the LIF to be DOWN because the system no longer has access to the PIFs.

## Examples

The following are examples of the ABORT SAC command:

```
ABORT SAC $ZZLAN.G11123.0
```

```
ABORT SAC ($ZZLAN.G11123.0, $ZZLAN.G11521.0)
```

```
ABORT SAC $ZZLAN.G11521.0, SUB
```



# ADD Command

ADD is a sensitive command that defines an object to the SLSA subsystem. The fully-qualified name assigned to the created object must be unique.

If an attribute for a given object is not assigned a value in the ADD command, the attribute's default value is used. However, some attributes do not have default values: use the ADD command to assign values.

The ADD command does not support wild-card characters.

## ADD ADAPTER Command

The ADD ADAPTER command assigns a name to a ServerNet adapter and also names the SACs and PIFs related to the specified adapter.

### Command Syntax

```
ADD [ /OUT file-spec/ ] ADAPTER adapter-name
    { , TYPE { E4SA|MIOE|TRSA|FESA|GESA|G4SA|CCSA|ATM3SA } }
    { , LOCATION (group,module,slot) }
    { , ACCESSLIST ( n0 , n1,..., n15 ) }
    [ , AUTODUMP { ON | OFF | EXTENDED } ]
    [ , AUTOFIRMUP { ON | OFF } ]
    [ , AUTOSTART { ON | OFF } ]
    [ , DLFILENAME file-spec ]
    [ , DUMPFILNAME file-spec ]
    [ , FIRMWAREFILENAME file-spec ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ADAPTER *adapter-name*

specifies the name to be assigned to the adapter. The names that the SLSA subsystem assigns to the SACs and PIFs are based on the name you assign to the adapter. For example, if you assign the name G11123 to a G4SA, the SAC on the adapter is named G11123.0. The PIF names are G11123.0.A, G11123.0.B, G11123.0.C, and G11123.0.D. SLSA treats the Ethernet ports in slots 6 and 7 of the VIO enclosure as G4SAs. (The VIO enclosure is available in H06.08 and subsequent H-series RVUs.)

TYPE { E4SA | MIOE | TRSA | FESA | GESA | G4SA | CCSA | ATM3SA }

specifies the type of adapter:

- E4SA specifies an Ethernet 4 ServerNet adapter (E4SA)
- MIOE specifies the Ethernet port on the multifunction I/O board (MFIOB)
- TRSA specifies the Token-Ring ServerNet adapter (TRSA)
- FESA specifies the Fast Ethernet ServerNet adapter
- GESA specifies the Gigabit Ethernet ServerNet adapter
- G4SA specifies the Gigabit Ethernet 4-port Ethernet ServerNet adapter or the Ethernet ports in the VIO enclosure.
- CCSA specifies the Common Communication ServerNet adapter
- ATM3SA specifies the ATM ServerNet adapter (ATM3SA)

LOCATION (*group, module, slot*)

specifies the group (system enclosure), module, and slot (physical, labeled space in a module) of the adapter you are adding to the SLSA subsystem. The LOCATION attribute identifies the adapter in terms of its physical location in the system enclosure. An adapter type of G4SA represents either an adapter in an IOAM enclosure or the Ethernet ports in a VIO enclosure (four Ethernet ports in a VIO enclosure correspond to a G4SA). In a VIO enclosure, the group is 100 and the slot is 6 or 7. For more information about the VIO enclosure, see the *Versatile I/O Manual*.

ACCESSLIST ( *n0* , *n1* , . . . , *n15* )

specifies the processors allowed to access the SAC subordinate to the specified adapter. The first processor in the list is the preferred owner for executing SCF commands. You can specify up to 16 processors in the access list. The processors do not have to be in the same cabinet as the adapter.

AUTODUMP { ON | OFF | EXTENDED }

specifies whether the SAC dumps information about SAC failure. This attribute does not apply to the MFIOB. The default is ON for all adapter types.

The AUTODUMP attribute can be specified as EXTENDED for the G4SA adapter type only. If this attribute is specified as ON, just the adapter data structures are dumped during an automatic or manual dump. If this attribute is set to EXTENDED, the data structures and I/O buffers are dumped. This attribute can be changed by using the ALTER SAC command even when the SAC is in the STARTED STATE. The EXTENDED option is recommended for use by HP support only. The default is ON.

AUTOFIRMUP { ON | OFF }

specifies whether the SAC automatically downloads the firmware prior to downloading the operation code. The firmware is downloaded if the version of the firmware file is newer than the firmware version on the adapter. The AUTOFIRMUP option does not apply to MFIOB adapters. The default is ON.

AUTOSTART { ON | OFF }

specifies whether the SAC and its subordinated PIFs automatically start after a system cold start. The default is ON.

DLFILENAME *file-spec*

specifies the name of the file that should contain the application microcode to be downloaded to the subordinated SACs when the SACs are started. This option does not apply to the MFIOB. The default is listed below.

DUMPFILNAME *file-spec*

specifies the name of the file in which dump information from the adapter is stored. Specify the numbers 00 for the last two characters of the file name. 00 allows the file name to be incremented if more than one dump file is needed. The DUMPFILNAME option does not apply to the MFIOB. The default is listed below.

FIRMWAREFILENAME *file-spec*

specifies the name of the file that contains the adapter firmware microcode to be downloaded to the SAC. This option does not apply to the MFIOB. The default is listed below.

## Considerations

- If the AUTODUMP attribute is set to EXTENDED, the dump size will be approximately 128 megabytes, which is much larger than the 64 megabyte dump size when AUTODUMP is selected with a setting of ON. The EXTENDED attribute is supported for G4SAs only.
- The ADAPTER object is not started when you use the ADD ADAPTER command to add the object to the subsystem and the ADAPTER object remains in the STOPPED state until a START ADAPTER command is issued. The subordinate SACs and PIFs also are in the STOPPED state and must be started.
- The specified ADAPTER name must be unique and not in use by any existing adapter objects. The name must be from one to eight alphanumeric characters long and use a leading alphabetic character.
- A TYPE, LOCATION, and ACCESSLIST must be specified.

- The following table lists the default values for DLFILENAME, FIRMWAREFILENAME and DUMPFILENAME:

Adapter	DLFILENAME	FIRMWAREFILENAME	DUMPFILENAME
ATM3SA	C0440P00	C8158R00	C0440D00
CCSA	C0303P00	C0309R00	C0303D00
E4SA	C7957P00	C7824R00	C7957D00
TRSA	C0084P00	C0232R00	C0084D00
FESA	C0301P00	C0283R00	C0301D00
GESA	C0507P00	C0506R00	C0507D00
G4SA	C0613P00	C0612R00	C0613D00

For MFIOB, the default file names are null.

- MFIOB, the default file names are null. Subordinate objects (SACs and PIFs) are created according to the specified TYPE.
- All subordinate SACs are created with the same attribute information (specified through ACCESSLIST, DLFILENAME, FIRMWAREFILENAME, DUMPFILENAME, AUTODUMP, AUTOFIRMUP, and AUTOSTART options). You must use the ALTER SAC command to give subordinate SACs different attribute values.
- You can specify only two processors in the access list when the type specified is MIOE.
- You cannot have duplicate processor numbers in the access list. The system returns a duplicate-number error message if you specify the same processor more than once.
- All subordinate PIFs have default attributes (see the [ALTER PIF Command](#) on page 4-32 for details).
- If SLSA E00008 is returned after issuing an ADD ADAPTER command, issue the following command to determine if another adapter is logically configured in the selected slot:

```
INFO ADAPTER $ZZSTO.*
```

## Examples

The following example assigns the name E0153 to an E4SA adapter type, residing in module 1, group 1, slot 53 of the system enclosure.

```
ADD ADAPTER $ZZLAN.E0153, TYPE E4SA, &  
LOCATION (1,1,53), ACCESSLIST (0,1,2,3)
```

The following example assigns the name E0152 to a FESA adapter type, residing in module 1, group 1, slot 52 of the system enclosure.

```
ADD ADAPTER $ZZLAN.F0152, TYPE FESA, &  
LOCATION (1,1,52), ACCESSLIST (0,1,2,3)
```

The following example assigns the name G0152 to a GESA adapter type, residing in module 1, group 1, slot 52 of the system enclosure.

```
ADD ADAPTER $ZZLAN.G0152, TYPE GESA, &  
LOCATION (1,1,52), ACCESSLIST (0,1,2,3)
```

The following example assigns the name G11123 to a G4SA adapter type, residing in Group 111, Module 2, and Slot 3 of an IOAM enclosure.

```
ADD ADAPTER $ZZLAN.G11123, TYPE G4SA, &  
LOCATION (111,2,3), ACCESSLIST (0,1,2,3)
```

The following example assigns the name M0IE0 to the Ethernet port on an MFIOB, residing in module 1, group 1, slot 50 of the system enclosure. Processor 0 has the primary access path to the adapter.

```
ADD ADAPTER $ZZLAN.M0IE0, TYPE MIOE, &  
LOCATION (1,1,50), ACCESSLIST (0,1)
```

The following example assigns the name T0254 to the TRSA, residing in module 2, group 1, slot 54 of the system enclosure. Processor 2 has the primary access path to the adapter.

```
ADD ADAPTER $ZZLAN.T0254, TYPE TRSA, &  
LOCATION (2,1,54), ACCESSLIST (2,3,0,1)
```

The following example assigns the name C0154 to an CCSA adapter type, residing in module 1, group 1, slot 54 of the system enclosure.

```
ADD ADAPTER $ZZLAN.C0154, TYPE CCSA, &  
LOCATION (1,1,54), ACCESSLIST (0,1,2,3)
```

The following example assigns the name ATM11 to an ATM3SA adapter type residing in module 2, group 1, slot 54 of the system enclosure.

```
ADD ADAPTER $ZZLAN.ATM11, TYPE ATM3SA, &  
LOCATION (2, 1, 54), ACCESSLIST (0, 1, 2, 3)
```

The following example assigns the name G11123 to the Ethernet ports residing in module 2, group 100, slot 6 of a VIO enclosure.

```
ASSUME PROCESS $ZZLAN
```

```
ADD ADAPTER G11123, type G4SA, location (100, 2, 6), access  
(0,1,2,3,4)
```

## ADD ATMSAP Command

The ADD ATMSAP command configures an ATMSAP object subordinate to a PIF object. The ATMSAP object is left in the STOPPED summary state if the command is successful. The command is rejected if the ATMSAP object name is already configured. The command is also rejected if the maximum ATMSAP objects subordinate to the PIF object have already been defined.

### Command Syntax

```
ADD [ /OUT file-spec/ ] ATMSAP atmsap-name
    [, MTU max-mtu-size ]
    [, NATURE { PVC } ]
    [, QOSSET { UBR } ]
    [, VCC (vpi,vci) ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ATMSAP *atmsap-name*

provides a unique name for the ATMSAP object that is being configured. *atmsap-name* is in the form

*\$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id*

where:

*adapter-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that identifies the parent SLSA ADAPTER. Wildcard characters are not allowed.

*sac-id*

is a number that identifies the parent SAC object. The only valid *sac-id* for the ATM3SA adapter is the number "0." Wildcard characters are not allowed.

*pif-id*

is an alpha character that identifies the parent PIF object. The only valid *pif-id* for the ATM3SA adapter is the letter "A." Wildcard characters are not allowed.

*atmsap-id*

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP object. Wildcard characters are not allowed.

MTU *max-mtu-size*

configures the maximum message transfer unit size for the ATMSAP.

Valid values:

*max-mtu-size* is a decimal number in the range 1 through 9180.

Default value: 9180

NATURE { PVC }

configures the virtual connection nature of the ATMSAP component and selects the connection supported for the ATMSAP object. Only PVC type connections are supported.

Valid values:

PVC: ATMSAP object uses a PVC connection

QOSSET { UBR }

configures the Quality of Service for the ATMSAP object. Only UBR is supported.

Valid values:

UBR: Unspecified bit rate.

VCC ( *vpi,vci* )

*vpi* configures the virtual path identifier. *vci* configures the virtual circuit identifier. This is a required parameter if NATURE PVC is selected.

Valid values:

Value	Description
<i>vpi</i>	Configures the virtual path identifier. Only vpi 0 is supported.
<i>vci</i>	Configures the virtual circuit identifier. The virtual circuit identifier is a number in the decimal range of 0 to 4096.

## Response Display

The ADD ATMSAP command returns only a success or failure indication. Successful completion is indicated when SCF displays the prompt for the next command. A failure is indicated when SCF displays an error message. In case of failure, additional information can be found in the EMS log.



## Considerations

- An ATMSAP object can be configured to interface to a permanent virtual circuit (PVC) object. A LIF object must be associated with the ATMSAP object in order to provide host access to the ATMSAP object. A PVC connection provides a static permanent virtual circuit. A PVC object is defined with a VCC attribute for the ATMSAP object. The VCC is comprised of a VPI, VCI pair.
- Use the ADD ATMSAP command to add an ATMSAP object. You can add a maximum of 128 ATMSAP object per PIF object.
- Use the ADD LIF command to configure a LIF object for each ATMSAP object.

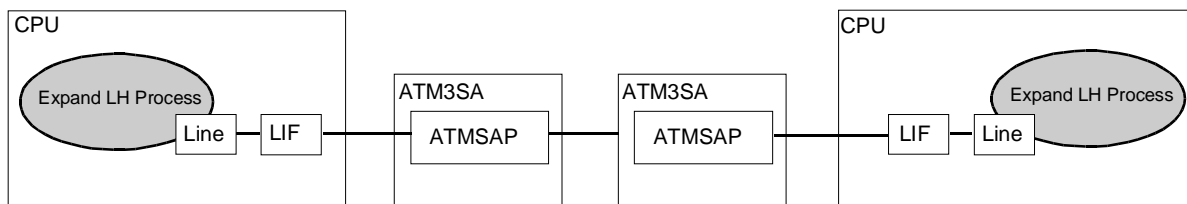
Following is an example of the ADD LIF command that adds a LIF object and associates it with an ATMSAP object.

```
ADD LIF $ZZLAN.LIF01, ATMSAP ATM0.0.A.ATMSAP01
```

The example above adds a LIF object named \$ZZLAN.LIF01 and associates it to an ATMSAP object named \$ZZLAN.ATM0.0.A.ATMSAP01.

- The ATMSAP object offers an ATM Native Mode network interconnect support similar to that offered by the PVC object within an ATM subsystem.

**Figure 4-2. Expand and ATMSAP**



VST016.vsd

## Example

```
ADD ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01, MTU 9180, NATURE PVC,  
QOSSET UBR, VCC(0,101),
```

## ADD LIF Command

The ADD LIF command configures a logical interface (LIF) into a system. The LIF must be associated with a media interface. For E4SA, FESA, GESA, G4SA, TRSA, and CCSA adapter types, a LIF object must be associated with the PIF object. For the ATM3SA adapter, a LIF object cannot be associated with the PIF object. Instead, it is associated with an ATMSAP object. You can configure up to 128 LIF objects to be associated with one ATM3SA ADAPTER. The LIF object is left in the STOPPED summary state if the ADD command is successful.

## Command Syntax

```
ADD [ /OUT file-spec/ ] LIF lif-name ,
{PIF pif-name | ATMSAP atmsap-name }
[ , DATAFORWARDMODE df-mode ]
[ , DATAFORWARDUNIT df-unit ]
[ , DATAFORWARDCOUNT df-count ]
[ , DATAFORWARDTIME df-time ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

LIF *lif-name*

specifies the name of the LIF to be added. *lif-name* is in the form \$ZZLAN.*lif-id* where

*lif-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that identifies the SLSA LIF object that is being configured. Wildcard characters are not allowed.

PIF *pif-name*

specifies the name of the PIF to be associated with the new LIF object. The PIF name has the form *adapter-name.sac-unit#.pif-unit-letter*, for example, G11123.0.A.

DATAFORWARDMODE {ADAPTIVE | MANUAL}

---

**Note.** HP recommends setting DATAFORWARDMODE to ADAPTIVE for G4SA adapters.

---

select ADAPTIVE to allow a G4SA adapter type to choose the best DATAFORWARDCOUNT (DFC) and DATAFORWARDTIME (DFT) values depending on data traffic. The G4SA automatically updates the DFC and DFT values whenever the data-traffic profile changes. The DATAFORWARDMODE ADAPTIVE value is supported for G4SA adapter types only.

Default value is MANUAL.

DATAFORWARDUNIT {MILLISECONDS | MICROSECONDS}

specifies the DFT values in MILLISECONDS or MICROSECONDS; MICROSECONDS is supported for G4SA adapter types only.

Default value is MILLISECONDS.

DATAFORWARDCOUNT *df-count*

specifies the maximum number of inbound frames queued before delivery occurs. *df-count* is the maximum number of frames queued.

Valid values:

*df-count* is a value in the range of 1 through 50.

Default value: 10 frames if the DATAFORWARDMODE value is MANUAL; otherwise, DATAFORWARDCOUNT defaults to 15 frames.

DATAFORWARDTIME *df-time*

configures the maximum time an inbound frame waits before delivery. *df-time* is the time to wait. This keyword is not valid when the LIF is being associated with an ATMSAP on an ATM3SA adapter. Instead, the keyword is supported by the ALTER PIF command.

*df-time* is specified as a time interval in the form *ttt*, where *ttt* is an integer where each unit is a millisecond or microsecond.

Default value: If DATAFORWARDMODE is MANUAL, DATAFORWARDTIME defaults to either 10 milliseconds or 10000 microseconds depending on the setting of DATAFORWARDUNIT. If DATAFORWARDMODE is ADAPTIVE, DATAFORWARDTIME defaults to 1 millisecond.

Range: If the DATAFORWARDUNIT is milliseconds, the *df-time* has a range of 1 through 200 milliseconds. If the DATAFORWARDUNIT is microseconds, the *df-time* has a range of 1 through 200000 microseconds.

## Considerations

- The specified LIF name must be unique and not in use by any existing SLSA LIF objects.
- The LIF name must begin with an alphabetic character and be no longer than eight alphanumeric characters.
- You must specify an existing PIF name or ATMSAP name, and no existing LIF can be already associated with the specified PIF or ATMSAP; otherwise, the ADD LIF command fails and the LIF object is not added to the SLSA subsystem.
- The LIF object is in the STOPPED state after you add it. You must start it by using the START LIF command.
- DATAFORWARDCOUNT and DATAFORWARDTIME can only be specified if DATAFORWARDMODE has been set to MANUAL. An error is returned if a DATAFORWARDMODE of ADAPTIVE or a DATAFORWARDUNIT of MICROSECONDS is specified for non-G4SA adapter types.
- When DATAFORWARDMODE is ADAPTIVE, the DATAFORWARDCOUNT and DATAFORWARDTIME settings are used as upper limits. An error is returned if a

DATAFORWARDUNIT of MICROSECONDS is specified for non-G4SA adapter types.

- DATAFORWARDCOUNT and DATAFORWARDTIME are meaningful only when the specified PIF is a part of an Ethernet 4 ServerNet adapter (E4SA), Fast Ethernet ServerNet adapter (FESA), Gigabit Ethernet ServerNet adapter (GESA), Gigabit Ethernet 4-port ServerNet Adapter (G4SA), or Token Ring ServerNet adapter (TRSA).

## Considerations When Using the DFC, DFM, and DFT Parameters

The Data Forward Count (DFC) and Data Forward Time (DFT) parameters are used to control the frame-forwarding behavior of the controller when sending incoming data to the host changing the frame forwarding behavior reduces the interrupt load on the host processor while retaining an acceptable forwarding delay.

Data frames are forwarded when the arriving frame count equals the configured DFC value or the configured DFT time has elapsed.

The challenge of configuring DFC and DFT is allowing for different data-traffic profiles. Normally DFC would be set to a high value to reduce the interrupt load on the host for a high-volume, streaming data profile, but this configuration may cause unacceptable forwarding delays for single-frame data traffic. Conversely, a DFC setting of 1 would provide immediate frame-forwarding but also incur the highest host-interrupt overhead because a host interrupt would be generated for every frame.

The Data Forward Mode (DFM) parameter provides a solution for configuring DFC/DFT for different traffic profiles. DFM can be set either to `MANUAL` or `ADAPTIVE`. If DFM is configured as `MANUAL`, DFC and DFT operate in the traditional manner as described above. If DFM is configured as `ADAPTIVE`, the frame-forwarding behavior adapts as the traffic profile changes.

In adaptive mode, DFC and DFT are configurable but interact differently. DFT represents the maximum time allowed before a group of frames are forwarded to the host. The DFC value, however, is dynamically adjusted based on the traffic pattern. Initially, DFC starts with a value of 1. This means that a data frame is forwarded to the host immediately upon reception providing the lowest forwarding delay. If frames continue to arrive intermittently, they are forwarded immediately. If the traffic pattern changes and a continuous frame stream is present, the DFC value increases, causing the interrupt load on the host to be reduced. The DFC stops increasing when it reaches the configured DFC value. The dynamic DFC value remains at the maximum as long as frames are forwarded based on DFC. If frame groups begin to time out and are forwarded based on the DFT value, the DFC is dynamically reduced to the point where frames are once again forwarded based on DFC. If the traffic pattern reverts to intermittent frame arrival, DFC returns to 1 providing minimal forwarding delay.

During some periods the traffic pattern may be changing so that data frames are forwarded based on DFT but, ideally, in adaptive mode, frames are mostly forwarded based on DFC, providing responsive forwarding characteristics along with a reduced host-interrupt load.

## Examples

The following example adds a LIF named L11123B to the system and associates it with PIF B under SAC 0 on G4SA G11123:

```
ADD LIF $ZZLAN.L11123B, PIF G11123.0.B
```

The following example changes the DATAFORWARDCOUNT value to five frames:

```
ADD LIF $ZZLAN.L11123B, PIF G11123.0.B, DATAFORWARDCOUNT 5
```

The following example changes the data forward mode to ADAPTIVE to allow a G4SA to choose the best DATAFORWARDCOUNT and DATAFORWARDTIME values depending on data traffic:

```
ADD LIF $ZZLAN.L11123B, PIF G11123.0.B, DATAFORWARDMODE  
ADAPTIVE
```

The following example changes the data forward unit to have a wait time in microseconds for a G4SA:

```
ADD LIF $ZZLAN.L11123B, PIF G11123.0.B, DATAFORWARDUNIT  
MICROSECONDS
```

## ALTER Command

ALTER is a sensitive command used to alter the attributes of an object.

### ALTER ATMSAP Command

The ALTER ATMSAP command reconfigures an ATMSAP object subordinate to a PIF object. The ATMSAP object must be in the STOPPED summary state if the command is to succeed.

### Command Syntax

```
ALTER [ /OUT file-spec/ ] ATMSAP atmsap-name  
      [, MTU max-mtu-size ]  
      [, NATURE { PVC } ]  
      [, QOSSET { UBR } ]  
      [, VCC (vpi,vci) ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ATMSAP *atmsap-name*

is the name of the target ATMSAP object. *atmsap-name* is in the form

*\$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id*

where:

*adapter-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that identifies the parent SLSA ADAPTER. Wildcard characters are not allowed.

*sac-id*

is a number that identifies the parent SAC object. The only valid *sac-id* for the ATM3SA adapter is the number "0." Wildcard characters are not allowed.

*pif-id*

is an alpha character that identifies the parent PIF object. The only valid *pif-id* for the ATM3SA adapter is the letter "A." Wildcard characters are not allowed.

*atmsap-id*

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP object. Wildcard characters are not allowed.

MTU *max-mtu-size*

*max-mtu-size* configures the maximum message transfer unit size for the ATMSAP object.

Valid values:

*max-mtu-size* is a decimal number in the range of 1 through 9180.

NATURE {PVC}

configures the virtual connection nature of the ATMSAP component. Only PVC is supported.

Valid values:

PVC: ATMSAP uses a PVC connection.

QOSSET { UBR }

configures the Quality of Service for the ATMSAP object. Only UBR is supported.

Valid Values:

UBR: Unspecified bit rate.

VCC (*vpi*,*vci*)

*vpi* configures the virtual path identifier. *vci* configures the virtual circuit identifier. This is a required parameter if NATURE PVC is selected.

Valid values:

Value	Description
<i>vpi</i>	Configures the virtual path identifier. Only <i>vpi</i> 0 is supported.
<i>vci</i>	Configures the virtual circuit identifier. The virtual circuit identifier is a number in the decimal range of 0 to 4096.

## Response Display

The ALTER ATMSAP command returns only a success or failure indication. Successful completion is indicated when SCF displays the prompt for the next command. A failure is indicated when SCF displays an error message.

## Example

```
ALTER ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01, MTU 9180, NATURE PVC,
QOSSET UBR, VCC (0, 101)
```

## ALTER LIF Command

The ALTER LIF command alters the data forward count (DFC) and data forward time (DFT). The LIF object must be in the STOPPED summary state for ALTER LIF to succeed.

## Command Syntax

```
ALTER [ /OUT file-spec/ ] LIF lif-name
[ , DATAFORWARDMODE df-mode ]
[ , DATAFORWARDUNIT df-unit ]
[ , DATAFORWARDCOUNT df-count ]
[ , DATAFORWARDTIME df-time ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

LIF *lif-name*

is the name of the LIF to be altered. The LIF name has the form \$ZZLAN.*lif-id*, where:

*lif-id*

is a 1 to 8-character alpha numeric string, beginning with an alpha character that identifies the target SLSA LIF object. Wildcard characters are not allowed.

DATAFORWARDMODE {ADAPTIVE | MANUAL}

---

**Note.** HP recommends setting DATAFORWARDMODE to ADAPTIVE for G4SA adapters.

---

select ADAPTIVE to allow a G4SA adapter type to choose the best DATAFORWARDCOUNT (DFC) and DATAFORWARDTIME (DFT) values depending on data-traffic. The G4SA automatically updates the DFC and DFT values whenever the data-traffic profile changes. The DATAFORWARDMODE ADAPTIVE value is supported for G4SA adapter types only.

Default value is MANUAL.

DATAFORWARDUNIT {MILLISECONDS | MICROSECONDS}

specifies the DFT values in MILLISECONDS or MICROSECONDS; MICROSECONDS is supported for G4SA adapter types only.

Default value is MILLISECONDS.

DATAFORWARDCOUNT *df-count*

specifies the maximum number of inbound frames queued before delivery occurs. *df-count* is the maximum number of frames queued.

This keyword is not valid when the LIF is being associated with an ATM3SA ADAPTER. Instead, the keyword is supported on the ALTER PIF command.

Valid values:

*df-count* is a value in the range of 1 through 50.

Default value: 10 frames if the DATAFORWARDMODE value is MANUAL; otherwise, DATAFORWARDCOUNT defaults to 15 frames.

DATAFORWARDTIME *df-time*

specifies the maximum time an inbound frame waits before delivery. *df-time* is the number of milliseconds to wait. This keyword is not valid when the LIF is being associated with an ATM3SA adapter. Instead, the keyword is supported on the ALTER PIF command. *df-time* is specified as a time interval in the form *ttt*, where:

*ttt*

is the number of milliseconds or microseconds.

Valid values:

*df-time* is a value in the range of 1 through 200 milliseconds or 1 through 2000000 microseconds when the DATAFORWARDUNIT value is set to microseconds.

For G4SA adapters using DATAFORWARDUNIT microseconds:

Valid values: a time interval in microseconds



Valid range: 1 to 200000 microseconds

Default value: If DATAFORWARDMODE is MANUAL, DATAFORWARDTIME defaults to either 10 milliseconds or 10000 microseconds depending on the setting of DATAFORWARDUNIT. If DATAFORWARDMODE is ADAPTIVE, DATAFORWARDTIME defaults to 1 millisecond.

## Considerations

- An ALTER LIF command is accepted only when the LIF object is in the STOPPED state.
- You must specify at least one option.
- An error is returned if a DATAFORWARDMODE of ADAPTIVE or a DATAFORWARDUNIT of MICROSECONDS is specified for non-G4SA adapter types.
- When DATAFORWARDMODE is ADAPTIVE, the DATAFORWARDCOUNT and DATAFORWARDTIME settings are used as upper limits. An error is returned if a DATAFORWARDUNIT of MICROSECONDS is specified for non-G4SA adapter types.
- DATAFORWARDCOUNT and DATAFORWARDTIME are meaningful only when the specified PIF is a part of an Ethernet 4 ServerNet adapter (E4SA), Fast Ethernet ServerNet adapter (FESA), Gigabit Ethernet ServerNet adapter (GESA), Gigabit Ethernet 4-port ServerNet adapter (G4SA), or Token Ring ServerNet adapter (TRSA).
- See [Considerations When Using the DFC, DFM, and DFT Parameters](#) on page 4-26

## Example

The following example changes the DATAFORWARDCOUNT (DFC) value for a specified LIF.

```
ALTER LIF $ZZLAN.L11123A , DATAFORWARDCOUNT 5
```

The following example changes the data forward mode to ADAPTIVE to allow a G4SA to choose the best DFC and DFT values depending on data-traffic:

```
ADD LIF $ZZLAN.L11123A, PIF G4SA.0.A, DATAFORWARDMODE  
ADAPTIVE
```

The following example changes the data forward unit to have a wait time in microseconds for a G4SA adapter:

```
ADD LIF $ZZLAN.L11123A, PIF G4SA.0.A, DATAFORWARDUNIT  
MICROSECONDS
```

## ALTER PIF Command

The ALTER PIF command alters Token Ring ServerNet adapters (TRSAs), Ethernet 4 ServerNet adapters (E4SAs), ATM ServerNet Adapters (ATM3SA), Fast Ethernet ServerNet adapters (FESAs), Gigabit Ethernet ServerNet adapters (GESAs), and Gigabit Ethernet 4-port ServerNet adapters (G4SAs), after they are added to the SLSA subsystem through the ADD ADAPTER command.

The ALTER PIF command only applies to the TRSA, E4SA, ATM3SA, FESA, GESA, and G4SA adapter types. The ALTER PIF command does not apply to the CCSA adapter type.

The AUTONEGOTIATION, DUPLEX, and LINESPEED attributes are only applicable for FESA, GESA, and G4SA PIFs. If these attributes are specified for non-FESA, non-GESA, or non-G4SA PIFs, the command is rejected with an error.

### Command Syntax

```
ALTER [ /OUT file-spec/ ] PIF pif-name [ , RESET ]
      [ , ACTIVEMONITOR { ON | OFF } ]
      [ , AUTONEGOTIATION { ON | OFF } ]
      [ , INTERFACE { AUTODETECT | COPPER | FIBER } ]
      [ , DATAFORWARDCOUNT df-count ]
      [ , DATAFORWARDTIME df-time ]
      [ , DUPLEX { HALF | FULL } ]
      [ , EARLYTOKENRELEASE { ON | OFF } ]
      [ , EMSVERBOSE { ON | OFF } ]
      [ , JUMBOFRAME { ON | OFF } ]
      [ , LINESPEED { 10 | 100 | 1000 } ]
      [ , MAXSESSIONS max-sessions ]
      [ , NODEMACADDRESS { addr | DEFAULT } ]
      [ , NOOPTION ]
      [ , RINGSPEED { 4 | 16 } ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PIF *pif-name*

is the name of the PIF to be altered. The PIF name has the form *\$ZZLAN.adapter-id.sac-id#.pifid*

where:

*adapter-id*

is a 1 to 8 character alpha-numeric string, beginning with an alpha character that identifies the parent ADAPTER objects. Wildcard characters are allowed.

*sac-id*

is a number that identifies the parent SAC objects. Wildcard characters are allowed.

*pif-id*

is an alpha character that identifies the target PIF objects. Wildcard characters are allowed.

ACTIVEMONITOR { ON | OFF }

specifies whether the adapter participates in the token-ring monitor function. Set it to ON for the node to participate as a standby or active monitor on the token-ring. The default is OFF. ACTIVEMONITOR is supported for TRSAs only.

AUTONEGOTIATION { ON | OFF }

Select ON to enable automatic speed negotiation from the FESA, GESA, or G4SA PIF. Select OFF to disable automatic speed negotiation from the FESA, GESA, or G4SA PIF. When AUTONEGOTIATION is set to OFF, you must also specify the DUPLEX and LINESPEED attributes. AUTONEGOTIATION is supported for FESAs, GESAs, and G4SAs only. The default is ON.

INTERFACE {AUTODETECT | COPPER | FIBER}

is supported for G4SA adapter types only. Select COPPER for PIF C or PIF D to use the 10/100/1000 copper ports. Select FIBER for PIF C or PIF D to use the 1000 fiber ports. Select AUTODETECT for PIF C or PIF D to automatically detect the interface type upon connection. (COPPER and AUTODETECT are the only possible settings for PIF A and PIF B.) The default is AUTODETECT.

DATAFORWARDCOUNT *df-count*

configures the maximum number of inbound frames queued before delivery occurs. *df-count* is the maximum number of frames queued. This attribute is supported for ATM3SAs only. The value of *df-count* is in the range of 1 through 50. The default value is 2. This attribute is supported for ATM3SAs only.

Valid values:

The range is 1 through 50.

Default value is 2.

DATAFORWARDTIME *df-time*

configures the maximum time an inbound frame waits before delivery. *df-time* is the time to wait. *df-time* is specified as a time interval in the form *ttt*, where *ttt* is an integer in which each unit is a millisecond. The value of *df-time* is in the range of 10 through 200 milliseconds. The default value is 10 milliseconds. This attribute is supported for ATM3SAs only. The value of *df-count* is in the range of 1 through 50. The default is 2. This attribute is supported for ATM3SAs only.

DUPLEX { HALF | FULL }

Select HALF to set the FESA, GESA, or G4SA communication line to half duplex. Select FULL to set the FESA, GESA, or G4SA communication line to full duplex. An error is returned if the AUTONEGOTIATION attribute is set to ON. The DUPLEX attribute is supported only for FESAs, GESAs, and G4SAs. The value must be set by the user. There is no default value.

EARLYTOKENRELEASE { ON | OFF }

specifies whether the adapter operates in the early token release mode. This parameter applies only to 16 Mbps operation. The default is OFF. EARLYTOKENRELEASE is supported for TRSAs only.

EMSVERBOSE { ON | OFF }

Select ON to enable issuing detailed informational EMS messages from the ATMSAP PIF. Select OFF to disable issuing detailed informational EMS messages from the ATM3SA PIF. Detailed informational EMS messages display control and flow of PIF processing within the ATM3SA adapter. This information is used primarily for debugging purposes. The default value is OFF. This attribute is supported for ATM3SAs only.

JUMBOFRAME { ON | OFF }

specifies whether the adapter operates with the IEEE standard of 1.5K bytes as the maximum transmission unit (MTU) or with the jumbo frame size of 9K bytes. The default is OFF. JUMBOFRAME is supported for GESAs and G4SAs only.

---

**Note.** Parallel Library TCP/IP is not supported on Integrity NonStop NS-series servers.

---

Jumbo frames are supported by Parallel Library TCP/IP and NonStop TCP/IPv6, but not by conventional TCP/IP. Parallel Library TCP/IP and NonStop TCP/IPv6 must be configured for use with jumbo frames. Use the ALTER SUBNET command to set the MTU value. Set an MTU size between 512 and 65535 in the configuration. To allow jumbo frames, set the MTU greater than the default frame size of 1500 bytes. Either the receiving PIF and all intermediate switches and routers must also have the same MTU size, or you must use PATHMTU to allow discovery of the MTU.

LINESPEED { 10 | 100 | 1000 }

Select 10 to set the FESA communication line to 10 Mbit/s. Select 100 to set the FESA communication line to 100 Mbit/s. The GESA and G4SA adapters with a copper interface can be set to any speed (typically 1000), but the fiber interface may only be set to 1000. An error is returned if the LINESPEED attribute is set when AUTONEGOTIATION is set to ON. The LINESPEED attribute is supported only for FESAs and GESAs. The value must be set by the user. There is no default value.

MAXSESSIONS *max-sessions*

*max-sessions* specifies the maximum sessions that the adapter can handle. The adapter pre-allocates resources to handle the sessions. The trade-off involved is between the number of sessions and the number of transmit-and-receive buffers for data. The range is 1 through 2000 sessions. The default is 2000.

NODEMACADDRESS { *addr* | DEFAULT }

*addr* specifies the primary MAC station address used by this node for all frames. DEFAULT specifies that the adapter gets the MAC address from the adapter SEEROM (the hardware MAC address). The range of values for *addr* is %H400000000000 to %H7FFFFFFFFFFFF.

For Token Ring, the valid non-default MAC address range is from %H400000000000 to %H7FFFFFFFFFFFF.

For Ethernet (E4SA, FESA, GESA, G4SA), the second hexadecimal digit in the MAC address must be 2, 6, A, or E to be a valid non-default MAC address. One of these values is needed to prevent the multicast bit from being set inadvertently. The address is read into memory by least significant bit (instead of most significant bit, like Token Ring); therefore, the last bit is read first, and must not be set, because the Ethernet standard requires that the first bit be the multicast bit.

The NODEMACADDRESS is valid for all adapter types, except the MIOE and CCSA adapters.

NOOPTION

is the dummy parameter used for PIFs on adapters other than TRSAs, E4SAs, FESAs, GESAs, G4SAs, and ATM3SAs. NOOPTION provides the INFO PIF, OBEYFORM command output for all other adapters. The NOOPTION parameter causes the SLSA subsystem to ignore the ALTER PIF command. The NOOPTION parameter cannot be used with any other parameter.

RINGSPEED { 4 | 16 }

specifies the speed of the token-ring. The speed is either 4 or 16 mbps. The default is 16. RINGSPEED is supported for TRSAs only.

## Considerations

- The PIF object must be in the STOPPED state before you can alter it.
- You must specify at least one attribute.
- The ALTER PIF command supports the ATM3SA, TRSA, E4SA, FESA, GESA, and G4SA PIFs. The ALTER PIF command does not apply to a CCSA since a CCSA does not have any alterable PIF attributes.
- If your token-ring LAN is using 3616 Token Ring adapters, check the ring speed before inserting TRSAs into your LAN. The TRSAs have a default ring speed of

16Mbps which could cause a conflict at ring-insertion time if there is a 3616 Token Ring adapter using a ring speed of 4Mbps.

- If PIF C is configured in AUTODETECT mode and both the copper and fiber C ports are connected to the network, the adapter code activates the fiber port and associate it with PIF C. Likewise for PIF D: if PIF D is configured in AUTODETECT mode and both the copper and fiber D ports are connected to the network, the adapter code activates the fiber port and associate it with PIF D.

## Examples

The following example uses the ALTER command to turn on active monitoring and to set the ring speed to 4.

```
ALTER PIF $ZZLAN.T0253.0.A , ACTIVEMONITOR ON, RINGSPEED 4
```

The following example uses the ALTER command to select the fiber ports on a G4SA adapter type.

```
ALTER PIF $ZZLAN.G11123.0.C, INTERFACE FIBER
```

## ALTER SAC Command

The ALTER SAC command alters the access list for the specified SAC and changes the names of the files for downloading and updating the firmware. The ALTER SAC command also sets the flag to specify whether the SAC dumps diagnostic information to a file if a failure occurs.

This command takes effect immediately.

## Command Syntax

```
ALTER [ /OUT file-spec/ ] SAC sac-name
{
  [ , ACCESSLIST ( n0 , n1, ..., n15 ) ]
  [ , AUTODUMP { ON | OFF | EXTENDED } ]
  [ , AUTOFIRMUP { ON | OFF } ]
  [ , AUTOSTART { ON | OFF } ]
  [ , DLFILENAME file-spec ]
  [ , FIRMWAREFILENAME file-spec ]
  [ , DUMPFILNAME file-spec ] } }
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

SAC *sac-name*

is the name of the SAC to be altered. The SAC name has the form \$ZZLAN.adapter-name.sac-unit#, for example, \$ZZLAN.G1123.0.

ACCESSLIST ( *n0* , *n1* , . . . , *n15* )

specifies the processors allowed to access the SAC. The first processor in the list is the preferred owner. You must specify at least two processors in the access list. The allowed number of processors in the access list is 2 to 16.

AUTODUMP { ON | OFF | EXTENDED }

specifies whether the SAC dumps information to a file if the SAC fails. AUTODUMP does not apply to the MIOE. AUTODUMP with the EXTENDED option only applies to the G4SA. AUTODUMP ON is the default. For the G4SA, if the attribute is ON, just the adapter data structures are dumped. If the attribute is set to EXTENDED, the data structures and I/O buffers are dumped. For the G4SA, this attribute can be changed even when the SAC is in the STARTED state. The EXTENDED option is recommended for use by HP support only.

AUTOFIRMUP { ON | OFF }

indicates whether the SAC automatically downloads the firmware prior to downloading the operation code if the version of the firmware file is newer than the firmware version on the adapter. ON indicates that the SAC automatically updates the firmware. OFF indicates the firmware is not updated automatically. The default is ON. The AUTOFIRMUP option does not apply to the MFIOB.

AUTOSTART { ON | OFF }

specifies that the system automatically starts the SAC after a system-load when this option is ON. If this option is set to OFF, you must start the SAC by using a START SAC command after the cold load. The default is ON.

DLFILENAME *file-spec*

specifies the file from which the SLSA subsystem obtains the application microcode to be downloaded to the SAC when the SAC changes from a STOPPED state to a STARTING state.

FIRMWAREFILENAME *file-spec*

specifies the name of the file that contains the firmware microcode to be loaded into the flash programmable ROM (PROM) of the specified SAC the next time the system console runs a FIRMUP command or AUTOFIRMUP occurs.

DUMPFILNAME *file-spec*

specifies the name of the file to which the adapter dumps diagnostic information when it detects a failure or when OSM or TSM initiates a dump of the adapter.

## Considerations

- To remove a processor from the access list or to add one, you must stop the SAC before executing the ALTER SAC command.
- You cannot have duplicate processor numbers in the access list. The system returns a duplicate-number error message if you specify the same processor more than once.
- For a SAC on an MFIOB, you can specify only two processors in the access list.
- If a SAC is in the STARTED or STARTING state, you can specify only the ACCESSLIST option, and the new access list must be a permutation of the current access list. If you want to add or remove a processor from the access list, stop the SAC before executing the ALTER SAC command.
- You cannot use wild-card characters with the ALTER command when you specify the ACCESSLIST option.
- You can alter the file names for DUMPFILENAME, DLFILENAME, and FIRMWAREFILENAME only when the SAC is in the STOPPED state.
- The default volume for all file names is \$SYSTEM.SYSnn.
- The SLSA subsystem does not check for the existence of the specified file and appropriate file contents until the file is used; for example, when a START SAC command is executed.
- A permutation of the access list while the SAC is in the STARTED state may cause an ownership change of the SAC.

## Examples

The following example uses the ALTER command to turn off the AUTODUMP attribute for a specified SAC:

```
ALTER SAC $ZZLAN.G11123.0, AUTODUMP OFF
```

The following example uses the ALTER command to change the access list for a specified SAC:

```
ALTER SAC $ZZLAN.G11123.0 , ACCESSLIST (2,3)
```

The following examples specify the name of a download file for a SAC, one on an E4SA, the others on a TRSA and a G4SA:

```
ALTER SAC $ZZLAN.E0153.1 , DLFILENAME C7957P00
```

```
ALTER SAC $ZZLAN.T0154.0 , DLFILENAME C0084P00
```

```
ALTER SAC $ZZLAN.G11123.0 , DLFILENAME C0613P00
```



# DELETE Command

DELETE is a sensitive command that removes objects from the subsystem.

## DELETE ADAPTER Command

The DELETE ADAPTER command removes an ADAPTER and its subordinate objects from the subsystem.

### Command Syntax

```
DELETE [ /OUT file-spec/ ] ADAPTER adapter-name
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ADAPTER *adapter-name*

is the name of the adapter to be deleted. The adapter name has the form \$ZZLAN.*adapter-name*, for example, \$ZZLAN.G11123.

### Considerations

- The ADAPTER object and its subordinate objects must be in the STOPPED state before you can delete the adapter. Also, all PIFs subordinate to the SAC on the adapter must not have any LIFs associated with them. All LIFs associated with PIFs or ATMSAPs subordinate to the specified ADAPTER object must be deleted before the specified ADAPTER object can be deleted. All ATMSAPs subordinated to the specified adapter must be deleted first.
- You cannot use a wild-card character in the name of the adapter when using the DELETE ADAPTER command.

### Example

The following example deletes the adapter named G11123 and its corresponding SACs and PIFs from the SLSA subsystem:

```
DELETE ADAPTER $ZZLAN.G11123
```

## DELETE ATMSAP Command

The DELETE ATMSAP command removes an ATMSAP object subordinate to a PIF object from the system configuration. The ATMSAP object must be in the STOPPED summary state if the command is to succeed. You must delete associated LIF objects before you can delete the ATMSAP object.

## Command Syntax

```
DELETE [ /OUT file-spec/ ] ATMSAP atmsap-name
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ATMSAP *atmsap-name*

is the name of the target ATMSAP object. *atmsap-name* is in the form

*\$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id*

where:

*adapter-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that identifies the parent SLSA ADAPTER. Wildcard characters are not allowed.

*sac-id*

is a number that identifies the parent SAC object. The only valid *sac-id* for the ATM3SA is the number "0." Wildcard characters are not allowed.

*pif-id*

is an alpha character that identifies the parent PIF object. The only valid *pif-id* for the ATM3SA adapter is the letter "A." Wildcard characters are not allowed.

*atmsap-id*

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP object. Wildcard characters are not allowed.

## Response Display

The DELETE ATMSAP command returns only a success or failure indication. Successful completion is indicated when SCF displays the prompt for the next command. A failure is indicated when SCF displays an error message.

## Example

```
DELETE ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01
```

## DELETE LIF Command

The DELETE LIF command removes a LIF from the subsystem.

## Command Syntax

```
DELETE [ /OUT file-spec/ ] LIF lif-name
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*LIF lif-name*

is the name of the LIF to be deleted. The LIF name has the form \$ZZLAN.*lif-name*, for example, \$ZZLAN.L11123B.

## Considerations

- The LIF must be in the STOPPED state and have no registered clients before it can be deleted.
- You cannot use a wild-card character in the name of the LIF when using the DELETE LIF command.

## Example

The following example deletes the LIF named L11123B from the SLSA subsystem:

```
DELETE LIF $ZZLAN.L11123B
```

## INFO Command

INFO is a nonsensitive command that displays the configured settings for the specified object. An asterisk (\*) next to a field in the display indicates that the attribute is alterable.

## INFO ADAPTER Command

The INFO ADAPTER command displays information about all adapters in a system or about a specific adapter.

## Command Syntax

```
INFO [/OUT file-spec/ ] ADAPTER adapter-name  
[ , DETAIL | OBEYFORM ]
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ADAPTER *adapter-name*

is the name of the adapter for which information is to be displayed.

DETAIL

causes the INFO command to display a list of detailed information about the adapter. Without this parameter, the command displays only type, group, module, and slot information.

OBEYFORM

causes the INFO command to display the information about the adapter in the form of ADD ADAPTER commands, so you can use the information to create the adapter's configuration.

## Considerations

- The INFO ADAPTER with the DETAIL option displays blank fields for board-specific information when the physical CRU or CRU information is missing.
- Access-list information is displayed when you specify the OBEYFORM option as place-holder information. For the actual access-list information in ACCESSLIST (and other SAC attributes), you must use INFO SAC, OBEYFORM.

## INFO ADAPTER Display

The display for the INFO ADAPTER command without the DETAIL option is:

```
-> INFO ADAPTER $ZZLAN.*

Name           Group Module Slot   Type
$ZZLAN.CC1      21      1     54   CCSA
$ZZLAN.E0154    1       1     54   E4SA
$ZZLAN.F0153    1       1     53   FESA
$ZZLAN.G11123   111     2      3   G4SA
$ZZLAN.M0IE1    1       1     55   MIOE
$ZZLAN.M0IE0    1       1     50   MIOE
```

The display for the INFO ADAPTER command with the DETAIL option is:

```
INFO ADAPTER $ZZLAN.G11123, DETAIL

SLSA Detailed Info ADAPTER \SYS.$ZZLAN.G11123

Board Partnum..... 527431-001
Board Revision..... A01-01
Board Serial Number.... MP0005
Board TNum..... LG4a
Group..... 110
Module..... 2
Slot..... 3
Type..... G4SA
```

Board PartNum

indicates the part number of the adapter.

Board Revision

indicates the revision level of the adapter.

Board Serial Number

indicates the unique serial number of the adapter.

Board TNum

indicates the HP product number of the adapter.

Group

indicates the group that the adapter resides in.

Module

indicates the module that the adapter resides in.

Slot

indicates the slot that the adapter resides in.

Type

indicates the type of the adapter.

The display for the INFO ADAPTER command with the OBEYFORM option is:

```
-> INFO ADAPTER $ZZLAN.* , OBEYFORM

ADD ADAPTER $ZZLAN.CC1 , &
    LOCATION (21 , 1 , 54) , &
    TYPE CCSA, &
    ACCESSLIST(0,1)
ADD ADAPTER $ZZLAN.F0153 , &
    LOCATION (1 , 1 , 53 ) , &
    TYPE FESA, &
    ACCESSLIST(0,1)
ADD ADAPTER $ZZLAN.E0154 , &
    LOCATION (1 , 1 , 54 ) , &
    TYPE E4SA, &
    ACCESSLIST(0,1)
ADD ADAPTER $ZZLAN.M0IE1 , &
    LOCATION (1 , 1 , 55 ) , &
    TYPE MIOE, &
    ACCESSLIST(0,1)
ADD ADAPTER $ZZLAN.M0IE0 , &
    LOCATION (1 , 1 , 50 ) , &
    TYPE MIOE, &
    ACCESSLIST(0,1)
```

## INFO ATMSAP Command

The INFO ATMSAP command returns the configuration information for an ATMSAP object subordinate to a PIF object.

### Command Syntax

```
INFO [ /OUT file-spec/ ] ATMSAP atmsap-name  
      [, {DETAIL | OBEYFORM} ]
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*ATMSAP atmsap-name*

is the name of the target ATMSAP object. *atmsap-name* is in the form

*\$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id*

where:

*adapter-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that identifies the parent SLSA ADAPTER. Wildcard characters are allowed.

*sac-id*

is a number that identifies the parent SAC object. The only valid *sac-id* for the ATM3SA adapter is the number “0.” Wildcard characters are allowed.

*pif-id*

is an alpha character that identifies the parent PIF object. The only valid *pif-id* for the ATM3SA adapter is the letter “A.” Wildcard characters are allowed.

*atmsap-id*

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP object. Wildcard characters are allowed.

DETAIL

causes the INFO ATMSAP command to display a list of detailed information.

OBEYFORM

causes the INFO ATMSAP command to display the information about the target ATMSAP objects in the form of ADD ATMSAP commands, so that you can recreate the ATMSAP object’s configuration.

## Response Display

The INFO ATMSAP command returns the configuration information for the target objects. There are three possible display formats for the ATMSAP information: summary, detailed, and obeyform. These formats are shown below. If no object can be found, the failure is indicated when SCF displays an error message.

Following is the display format for the ATMSAP INFO command with the DETAIL option not selected.

```
1-> info ATMSAP $zzlan.atm1.0.a.atmsap01

SLSA Info ATMSAP

Name                Nature      VCC      MTU
$ZZLAN.ATM1.0.A.ATMSAP01  PVC      (0,100)  9180
```

Following is the display format for the ATMSAP INFO command with the DETAIL option selected:

```
1-> info atmsap $zzlan.atm1.0.a.atmsap01,detail

SLSA Detailed Info ATMSAP for \SYS.$ZZLAN.ATM1.0.A.atmsap01

*MTU          9180
*NATURE       PVC
*QOSSET       UBR
*VCC          (0,100)
```

Following is the display format for the ATMSAP INFO command with the OBEYFORM option selected:

```
1-> info ATMSAP $zzlan.atm1.0.a.atmsap01,obeyform

ADD ATMSAP $ZZLAN.ATM1.0.A.ATMSAP01, &
MTU      9180      , &
NATURE   PVC       , &
QOSSET   UBR       , &
VCC      (0,100)
```

MTU

indicates the maximum message transfer unit size for the ATMSAP object.

NATURE

indicates the virtual connection nature of the ATMSAP component.

QOSSET

indicates the Quality of Service for the ATMSAP object.

VCC ( *vpi*, *vci* )

indicates the configured virtual path identifier (*vpi*) and configured virtual circuit identifier (*vci*) of the ATMSAP object.

## Example

```
INFO ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01
INFO ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01, DETAIL
INFO ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01, OBEYFORM
```

## INFO LIF Command

The INFO LIF command displays the current attribute setting for the specified LIF.

### Command Syntax

INFO [ /OUT <i>file-spec</i> / ] LIF <i>lif-name</i> [ , DETAIL   OBEYFORM]
---

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

LIF *lif-name*

is the name of the LIF for which information is to be displayed.

DETAIL

causes the INFO command to display detailed information about the LIF; otherwise, only MAC addresses, associated PIFs, and PIF types are displayed.

OBEYFORM

causes the INFO command to display the information about the adapter in the form of ADD LIF commands.

### Considerations

The INFO LIF command is not supported for MFIOB adapters.



## INFO LIF Displays

The display for the INFO LIF command without the DETAIL option is:

```
1-> INFO LIF $ZZLAN.*
SLSA Info LIF
```

Name	Associated Object	MAC Address	Type
\$ZZLAN.CC10A	CC1.0.A	00:00:00:00:00:00	WAN
\$ZZLAN.LANY	M0IE1.0.A	08:00:8E:00:7A:D9	Ethernet
\$ZZLAN.LANX	M0IE1.0.A	08:00:8E:00:7B:BA	Ethernet
\$ZZLAN.L018	E0153.0.A	08:00:8E:00:78:3A	Ethernet
\$ZZLAN.L019	E0153.0.B	08:00:8E:00:78:2D	Ethernet
\$ZZLAN.L01A	E0153.1.A	08:00:8E:00:78:2C	Ethernet
\$ZZLAN.L01B	E0153.1.B	08:00:8E:00:78:1C	Ethernet
\$ZZLAN.L11123A	G11123.0.A	08:00:8E:00:97:B0	Ethernet
\$ZZLAN.L01C	T0154.0.A	08:00:8E:80:12:CE	Token Ring
\$ZZLAN.L01A	E0152.0.A	08:00:8E:AB:CD:EF	Ethernet

The display for the INFO LIF command with the DETAIL option is:

```
1-> INFO LIF $ZZLAN.G11123, DETAIL

SLSA Detailed Info LIF \SYS.$ZZLAN.G11123

*DataForwardMode..... ADAPTIVE
*DataForwardUnit..... MILLISECONDS
*DataForwardCount..... 15
*DataForwardTime..... 1
  LIF Type..... Ethernet
  MAC Address..... 08:00:8E:00:97:B0
  PIF Name..... G11123.0.A
```

### DataForwardMode

---

**Note.** HP recommends setting DATAFORWARDMODE to ADAPTIVE for G4SA adapters.

---

is either ADAPTIVE or MANUAL. ADAPTIVE allows a G4SA adapter type to choose the best DATAFORWARDCOUNT and DATAFORWARDTIME values depending on data traffic and to automatically update these values when the traffic profile changes. ADAPTIVE is supported for G4SA adapter types only. MANUAL allows a user to set the DFC and DFT values, and the INFO LIF command with the detail option reflects these settings.

DataForwardUnit {MILLISECONDS | MICROSECONDS}

specifies the DFT values in MILLISECONDS or MICROSECONDS; MICROSECONDS is supported for G4SA adapter types only.

Default value is MILLISECONDS.

DataForwardCount

specifies the maximum number of inbound frames queued before delivery occurs.

DataForwardTime

specifies the maximum time an inbound frame waits before delivery.

LIF Type

indicates the type of network interface used.

MAC Address

indicates the MAC address of the given LIF (used by the associated PIF).

PIF Name

indicates the name of the PIF associated with the LIF.

The display for the INFO LIF command with the OBEYFORM option is:

```
-> INFO LIF $ZZLAN.* , OBEYFORM

ADD LIF $ZZLAN.LANY , &
    PIF M0IE1.0.A, &
    DATAFORWARDCOUNT 10 , &
    DATAFORWARDTIME 10
ADD LIF $ZZLAN.LANX , &
    PIF M0IE0.0.A, &
    DATAFORWARDCOUNT 10 , &
    DATAFORWARDTIME 10
ADD LIF $ZZLAN.E0152 , &
    PIF E0152.0.A, &
    DATAFORWARDCOUNT 10 , &
    DATAFORWARDTIME 10
ADD LIF $ZZLAN.E0153 , &
    PIF E0153.1.A, &
    DATAFORWARDCOUNT 10 , &
    DATAFORWARDTIME 10
ADD LIF $ZZLAN.E0153 , &
    PIF E0153.0.A, &
    DATAFORWARDCOUNT 10 , &
    DATAFORWARDTIME 10
ADD LIF $ZZLAN.E0153 , &
    PIF E0153.1.B, &
    DATAFORWARDCOUNT 10 , &
    DATAFORWARDTIME 10
ADD LIF $ZZLAN.E0153 , &
    PIF E0153.0.B, &
    DATAFORWARDCOUNT 10 , &
    DATAFORWARDTIME 10
```

# INFO MON Command

The INFO MON command displays the attribute setting for the specified LAN Monitor (LANMON) process.

## Command Syntax

```
INFO [ /OUT file-spec/ ] MON lanmon-name [ , DETAIL ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

MON *lanmon-name*

is the name of the LANMON process for which information is to be displayed. If you do not specify a monitor process, the SCF subsystem displays information for all monitor processes in the system.

DETAIL

causes the INFO command to display a list of detailed information.

## Considerations

The INFO MON command without the DETAIL option displays the same information as the INFO MON command with the DETAIL option.

## INFO MON Display

```
-> INFO MON $ZZLAN.*

SLSA Info MON

Name          CPU    SaveAbend  Program Name
$ZZLAN.#ZLM00    1      ON      \SYS.$SYSTEM.SYS04.LANMON
$ZZLAN.#ZLM01    0      ON      \SYS.$SYSTEM.SYS04.LANMON
$ZZLAN.#ZLM02    2      ON      \SYS.$SYSTEM.SYS04.LANMON
$ZZLAN.#ZLM03    3      ON      \SYS.$SYSTEM.SYS04.LANMON
```

CPU

indicates the processor in which the LANMON process is running.

SaveAbend

indicates whether the SAVEABEND flag of the process is on or off.

Program Name

indicates the name of the program file used by the LANMON process.

## INFO PIF Command

The INFO PIF command displays the physical interface (PIF) type and information about the attributes of the PIF.

### Command Syntax

```
INFO [ /OUT file-spec/ ] PIF pif-name [ , DETAIL | OBEYFORM]
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*PIF pif-name*

is the name of the PIF for which information is to be displayed. If you do not specify a PIF, the SCF subsystem displays information for all PIFs. The PIF name has the form `$ZZLAN.adapter-name.sac-unit#.pif-unit-letter`, for example, `$ZZLAN.G11123.0.A`.

*DETAIL*

causes the INFO command to display a list of detailed information about the PIF. Without this option, the command displays only the name, MAC address, and type information.

*OBEYFORM*

causes the INFO command to display the information about the adapter in the form of ALTER PIF commands so you can recreate the adapter's configuration.

### Considerations

- If no physical adapter exists or the PIF has not been STARTED, the display shows 00:00:00:00:00:00 for the MAC address.
- PIF types are not supported on MFIOB adapters.
- TRSA, E4SA, ATM3SA, FESA, GESA, and G4SA adapter types have PIF attributes that you can alter. (See the [ALTER PIF Command](#) on page 4-32.)

## INFO PIF Display

The attributes displayed for INFO PIF are the same as those that can be altered with the ALTER PIF command. See ALTER PIF for definitions of the attributes.

The display for the INFO PIF command with neither the DETAIL nor OBEYFORM option selected is:

```
1>INFO PIF $ZZLAN.*

SLSA Info PIF \ROO.$ZZLAN

Name                               NodeMAC Address          Type
$ZZLAN.E0153.0.A                  08:00:8E:00:5E:F7        Ethernet
$ZZLAN.E0153.0.B                  08:00:8E:00:5E:F8        Ethernet
$ZZLAN.E0153.1.A                  08:00:8E:00:5E:F9        Ethernet
$ZZLAN.E0153.1.B                  08:00:8E:00:5E:FA        Ethernet
$ZZLAN.G11123.0.A                 00:04:76:DC:0B:51        Ethernet
$ZZLAN.G11123.0.B                 00:04:76:DC:0B:52        Ethernet
$ZZLAN.G11123.0.C                 00:04:76:DC:0B:53        Ethernet
$ZZLAN.G11123.0.D                 00:04:76:DC:0B:54        Ethernet
$ZZLAN.A0253.0.A                  N/A                       ATM
```

## INFO PIF DETAIL Display For ATM3SA Adapters

The display format for INFO PIF with DETAIL selected for ATM3SAs is:

```
1> INFO PIF $ZZLAN.a0253.0.a.detail

SLSA Detailed Info PIF \ROO.$ZZLAN.A0253.0.A

Interface Speed..... 155Mbit/sec
PIF Type..... ATM

ATM3SA Specific Info

*DATAFORWARDCOUNT..... 2
*DATAFORWARDTIME..... 1
*EMSVERBOSE..... OFF
```

DATAFORWARDCONT *df-count*

indicates the configured maximum number of inbound frames queued before delivery occurs

DATAFORWARDTIME *df-time*

indicates the configured maximum time an inbound frame waits before delivery.

EMSVERBOSE { ON | OFF }

indicates whether or not detailed informational EMS messages are enabled.

The display for the INFO PIF command with the OBEYFORM option for ATM3SAs is:

```
-> INFO PIF $ZZLAN.ATM1.0.A.OBEYFORM
ALTER PIF $ZZLAN.A0253.0.A &
      ,DATAFORWARDCOUNT 2 &
      ,DATAFORWARDTIME 1 &
      ,EMSVERBOSE OFF
```

## INFO PIF DETAIL Display for an E4SA

The display for the INFO PIF command with the DETAIL option is:

```
1>INFO PIF $ZZLAN.E0153.0.A , DETAIL

SLSA Detailed Info PIF \SYS.$ZZLAN.E0153.0.A

Hardware MAC Address.... 08:00:8E:00:78:32
Interface Speed..... 10Mbit/sec
Max Frame Size..... 1514
Min Frame Size..... 60
*NodeMACAddress..... 08:00:8E:00:78:32
PIF Type..... Ethernet
```

Hardware MAC Address

indicates the default MAC address of the specified PIF.

Interface Speed

indicates the maximum speed of the network interface.

Max Frame Size

indicates the maximum frame size that the adapter can handle.

Min Frame Size

indicates the minimum size frame that the adapter can handle.

NodeMACAddress

indicates the primary MAC address used by this node for all transmitted and received frames. If the node MAC address is not supplied, the default MAC address from the SEEROM on the adapter is used (that is, the hardware MAC address). The high-order bit must be 0 and the next highest bit must be 1 for a locally administered address.

PIF Type

indicates the type of network interface.

## INFO PIF DETAIL Display for a FESA

The display for the INFO PIF command with the DETAIL option for FESAs is:

```
1>SCF INFO PIF $ZZLAN.F0253.0.a.detail

SLSA Detailed Info PIF \SYS.$ZZLAN.F0253.0.A

Hardware MAC Address.... 08:00:8E:00:90:13
Interface Speed..... 100Mbit/sec
Max Frame Size..... 1516
Min Frame Size..... 60
PIF Type..... Ethernet

FESA Specific Info

*AUTONEGOTIATION..... OFF
*DUPLEX..... HALF
*LINESPEED..... 100
```

### Hardware MAC Address

indicates the default MAC address from the SEEROM on the adapter.

### Interface Speed

indicates the maximum speed of the network interface.

### Max Frame Size

indicates the maximum frame size that the adapter can handle.

### Min Frame Size

indicates the minimum frame size that the adapter can handle.

### PIF Type

indicates the type of network interface.

### Autonegotiation

indicates whether the sender and receiver can negotiate a line speed. If Autonegotiation is ON, you may not alter the duplex value or linespeed value.

### Duplex

indicates full-duplex operation or half-duplex operation.

### Linespeed

indicates the line speed.

The INFO PIF display with AUTONEGOTIATION OFF for a FESA is:

```
1>SCF INFO PIF \SYS.$ZZLAN.F0154.0.1, DETAIL
SLSA Detailed Info PIF \SYS.$ZZLAN.F0154.0.1
  Hardware MAC Address.... 08:00:8E:00:DD:FB
*Autonegotiation.....OFF
*Duplex.....FULL
*Linespeed.....100 Mbit/sec
  Max Frame Size..... 1516
  Min Frame Size..... 60
*NodeMACAddress..... 08:00:8E:00:DD:FB
  PIF Type..... Ethernet
```

#### Hardware MAC Address

indicates the default MAC address from the SEEROM on the adapter.

#### Autonegotiation

indicates the status of automatic speed negotiation.

#### Duplex

indicates the setting of the duplex attribute of the FESA communication line.

#### Linespeed

indicates the setting of the FESA communication line.

#### Max Frame Size

indicates the maximum frame size that the adapter can handle.

#### Min Frame Size

indicates the minimum size frame that the adapter can handle.

#### NodeMACAddress

indicates the primary MAC address used by this node for all transmitted and received frames. If the node MAC address is not supplied, the default MAC address from the SEEROM on the adapter is used (that is, the hardware MAC address). The high-order bit must be 0 and the next highest bit must be 1 for a locally administered address.

#### PIF Type

indicates the type of network interface.



The INFO PIF display with AUTONEGOTIATION ON for a FESA is:

```
1>SCF INFO PIF \SYS.$ZZLAN.F0154.0.A, DETAIL
SLSA Detailed Info PIF \SYS.$ZZLAN.F0154.0.A
  Hardware MAC Address.... 08:00:8E:00:DD:FB
*Autonegotiation.....ON
*Duplex.....
*Linespeed.....
  Max Frame Size..... 1516
  Min Frame Size..... 60
*NodeMACAddress..... 08:00:8E:00:DD:FB
  PIF Type..... Ethernet
```

#### Hardware MAC Address

indicates the default MAC address from the SEEROM on the adapter.

#### Autonegotiation

indicates the status of automatic speed negotiation.

#### Duplex

indicates the setting of the duplex attribute of the FESA communication line.

#### Linespeed

indicates the setting of the FESA communication line.

#### Max Frame Size

indicates the maximum frame size that the adapter can handle.

#### Min Frame Size

indicates the minimum size frame that the adapter can handle.

#### NodeMACAddress

indicates the primary MAC address used by this node for all transmitted and received frames. If the node MAC address is not supplied, the default MAC address from the SEEROM on the adapter is used (that is, the hardware MAC address). The high-order bit must be 0 and the next highest bit must be 1 for a locally administered address.

#### PIF Type

indicates the type of network interface.

## INFO PIF DETAIL Display for a GESA

The display format for INFO PIF with DETAIL selected for GESAs is:

```
1>INFO PIF $ZZLAN.G0253.0.A , DETAIL

SLSA Detailed Info PIF \SYS.$ZZLAN.G0253.0.A

  Hardware MAC Address.... 08:00:8E:00:78:B0
  Interface Speed..... 1000 Mbit/sec
  Max Frame Size..... 9014
  Min Frame Size..... 60
  *NodeMACAddress..... 08:00:8E:00:78:B0
  PIF Type..... Ethernet

GESA Specific Info

*AutoNegotiation.....ON
*Duplex .....
*LineSpeed .....
*JumboFrame .....ON
  Interface Type .....Copper
```

### Hardware MAC Address

indicates the default MAC address from the SEEROM on the adapter.

### Interface Speed

indicates the maximum speed of the network interface.

### Max Frame Size

indicates the maximum frame size that the adapter can handle. This display shows a setting for jumbo frames.

### Min Frame Size

indicates the minimum frame size that the adapter can handle.

### NodeMACAddress

indicates the primary MAC address used by this node for all transmitted and received frames. If the node MAC address is not supplied, the default MAC address from the SEEROM on the adapter is used (that is, the hardware MAC address). The high-order bit must be 0 and the next highest bit must be 1 for a locally administered address.

### PIF Type

indicates the type of network interface.

### Autonegotiation

indicates whether the sender and receiver can negotiate a line speed. If Autonegotiation is ON, you may not alter the duplex value or line speed value.

**Duplex**

indicates full-duplex or half-duplex operations. The value is not shown, because Autonegotiation is ON.

**Linespeed**

indicates the line speed. Value not shown, because Autonegotiation is ON.

**JumboFrame**

indicates the choice of normal or jumbo frames (activated by the Maximum Transmission Unit (MTU) setting in your Parallel Library TCP/IP or NonStop TCP/IPv6 configuration).

---

**Note.** Parallel Library TCP/IP is not supported on Integrity NonStop NS-series servers.

---

**Interface Type**

indicates the physical cabling: copper or short-haul fiber (SX).

**INFO PIF DETAIL Display for a G4SA**

The display format for INFO PIF with DETAIL selected for G4SAs is:

```
1>INFO PIF $ZZLAN.G11123.0.A , DETAIL

SLSA Detailed Info PIF \SYS.$ZZLAN.G11123.0.A

  Hardware MAC Address.... 00:04:76:DC:0B:51
  Interface Speed..... 1000 Mbit/sec
  Max Frame Size..... 1514
  Min Frame Size..... 60
  *NodeMACAddress..... 00:04:76:DC:0B:51
  PIF Type..... Ethernet

G4SA Specific Info

*AutoNegotiation.....ON
*Duplex .....
*Linespeed .....
*JumboFrame .....OFF
*Interface .....AUTODETECT
```

**Hardware MAC Address**

indicates the default MAC address from the SEEROM on the adapter.

**Interface Speed**

indicates the maximum speed of the network interface.

**Max Frame Size**

indicates the maximum frame size that the adapter can handle.

### Min Frame Size

indicates the minimum frame size that the adapter can handle.

### NodeMACAddress

indicates the primary MAC address used by this node for all transmitted and received frames. If the node MAC address is not supplied, the default MAC address from the SEEROM on the adapter is used (that is, the hardware MAC address). The high-order bit must be 0 and the next highest bit must be 1 for a locally administered address.

### PIF Type

indicates the type of network interface.

### Autonegotiation

indicates whether the sender and receiver can negotiate a line speed. If Autonegotiation is ON, you may not alter the duplex value or line-speed value.

### Duplex

indicates full-duplex or half-duplex operations. The value is not shown because Autonegotiation is ON.

### Linespeed

indicates the line speed. The value is not shown because Autonegotiation is ON.

---

**Note.** Parallel Library TCP/IP is not supported on Integrity NonStop NS-series servers.

---

### JumboFrame

indicates the choice of normal or jumbo frames (activated by the Maximum Transmission Unit (MTU) setting in your Parallel Library TCP/IP or NonStop TCP/IPv6 configuration).

### Interface

indicates the interface type: copper, fiber, or automatic detection upon connection.

## INFO PIF DETAIL Display for a TRSA

The display for the INFO PIF command with the DETAIL option for TRSAs is:

```
1>SCF INFO PIF \SYS.$ZZLAN.T0154.0.A, DETAIL

SLSA Detailed Info PIF \SYS.$ZZLAN.T0154.0.A

Hardware MAC Address.... 08:00:8E:00:DD:FB
Interface Speed..... N/A
Max Frame Size..... 17832
Min Frame Size..... 32
*NodeMACAddress..... 08:00:8E:00:DD:FB
PIF Type..... Token Ring

TRSA Specific Info

*ActiveMonitor..... OFF
*EarlyTokenRelease.... OFF
*MaxSessions..... 2000
*RingSpeed..... 16 Mb
```

### Hardware MAC Address

indicates the default MAC address from the SEEROM on the adapter.

### Interface Speed

indicates the maximum speed of the network interface.

### Max Frame Size

indicates the maximum frame size that the adapter can handle.

### Min Frame Size

indicates the minimum frame size that the adapter can handle.

### NodeMACAddress

indicates the primary MAC address used by this node for all transmitted and received frames. If the node MAC address is not supplied, the default MAC address from the SEEROM on the adapter is used (that is, the hardware MAC address). The high-order bit must be 0 and the next highest bit must be 1 for a locally administered address.

### PIF Type

indicates the type of network interface.

### ActiveMonitor

specifies whether the adapter participates in the monitor function of token-ring LANs. ON specifies that the node is either a standby monitor or an active monitor on the ring. The token-ring chipset handles the monitor and contention protocol, and switches from standby monitor to active monitor if required. All token-ring LANs need an active monitor. The default value is OFF.

### EarlyTokenRelease

specifies whether the adapter operates in Early Token Release mode (16 Mbps operation only).

### MaxSessions

is the maximum number of sessions that can be added to the adapter by all I/O process methods. Currently, sessions can only be added by the SNATR MSAP. Use this attribute to tune the adapter to match the most-common maximum number of sessions being handled. The adapter allocates the resources for MaxSessions; the smaller the value, the more transmit and receive buffers the adapter has available for data.

### RingSpeed

specifies the speed at which to run the token ring. Specify either 4 or 16 Mbps.

## INFO SAC Command

The INFO SAC command displays configurable and non-configurable information about the specified SAC.

### Command Syntax

INFO [ /OUT <i>file-spec</i> / ] SAC <i>sac-name</i> [ , DETAIL   OBEYFORM ]
--

#### OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

#### SAC *sac-name*

is the name of the SAC for which information is to be displayed. The SAC name has the form \$ZZLAN.*adapter-name.sac-unit*#, for example, \$ZZLAN.G11123.0.

#### DETAIL

causes the INFO command to display a list of detailed information.

#### OBEYFORM

causes the INFO command to display the information about the SAC in the form of ALTER SAC commands, so that you can recreate the adapter's configuration.

## Considerations

- The ServerNet identification (SvNet ID) is reset to 0x00000000 when CRU information is not available or when the physical CRU is missing.
- The firmware file version, firmware revision, and download version cannot be displayed for the Ethernet port on the multifunction I/O board (MFIOB).
- SvNet ID field is reset to 0x00000000 when CRU information is not available or physical CRU is missing.

## INFO SAC Display

The display for the INFO SAC command without the DETAIL option is:

```
1>SCF INFO SAC $ZZLAN.E0*

SLSA Info SAC

Name                Owner  *Access List
$ZZLAN.CC1.0        2      (2,3,1,0,11,10,13,12,9,8,4,5,6,7,14,15)
$ZZLAN.E0153.0      3      (3,2,1,0)
$ZZLAN.E0153.1      2      (2,3,1)
$ZZLAN.E0154.0      0      (0,1)
$ZZLAN.E0154.1      1      (1,0)
$ZZLAN.G11123.0     0      (0,1)
```

Owner

indicates the processor that currently has the primary data path to the SAC.

Access List

shows the processors that can have data paths to the SAC.

The display for the INFO SAC command with the DETAIL option is:

```
-> INFO SAC $ZZLAN.G11123.0 , DETAIL

SLSA Detailed Info SAC \SYS.$ZZLAN.G11123.0

*AutoFirmup..... ON
  AutoReload..... ON
*AutoStart..... ON
*AutoDump..... ON
*AccessList..... ( 0, 1, 2, 3 )
*DlFilename..... C0613P00
*FirmwareFilename..... C0612R00
*DumpFilename..... C0613D00
  SAC Type..... G4SA SAC
  Owner CPU..... 2
  SvNet ID..... 0x000E00D2
  Download Version..... T0613G06_04JUN2004_04JUN2004
  Firmware Revision..... T0612G06_04JUN2004_04JUN2004
  Firmware File Rev..... T0612G06_04JUN2004_04JUN2004
```

### AutoFirmup

indicates whether the firmware revision file and the firmware currently in the SAC are compared during the startup sequence. The firmware revision file and the current firmware are always compared but not automatically updated. ON indicates that the firmware file is downloaded to the adapter if the firmware file is a newer revision than that currently running in the adapter. The default is ON. The AUTOFIRMUP option does not apply to MFIOB adapters.

### AutoReload

indicates whether application microcode is downloaded automatically to the SAC if it fails. ON indicates that the code is downloaded. OFF indicates that no application microcode is downloaded. The default is ON.

### AutoStart

indicates whether the SAC automatically starts after a system cold start. OFF indicates that the SAC and its subordinate PIFs are left in the STOPPED state after a cold start. ON indicates that the SAC and its subordinate PIFs automatically start after a cold start. The default is ON.

### AutoDump

indicates whether the SAC attempts to dump the contents of the adapter's memory before it restarts after a failure of the SAC object. OFF indicates that the SAC does not dump the memory contents automatically. ON indicates that the SAC dumps the information before it restarts. The default is ON.

### AccessList

lists the processors that can have access to the SAC.

### DlFilename

specifies the name of the file that contains the application microcode for the SAC.

### FirmwareFilename

indicates the name of the file that contains the firmware microcode.

### DumpFilename

indicates the name of the file to which the SAC dumps the contents of the adapter's memory if the SAC object fails.

### SAC Type

indicates the type of SAC in use. The type is specified as either MIOE SAC, E4SA SAC, FESA SAC, GESA SAC, G4SA, or TRSA SAC. A value of Adapter Mismatch means that the SAC has been configured for an adapter type that differs from the



actual adapter type that's in the slot. A value of No Adapter means that there is no adapter in the slot.

Owner CPU

indicates the processor that currently has ownership of the SAC.

SvNet ID

indicates the ServerNet ID of the SAC.

Download Version

indicates the version of the application microcode information.

Firmware Revision

indicates the version of the firmware code currently running on the SAC.

Firmware File Rev

indicates the current revision level of the firmware code, which is in the firmware file that is downloaded the next time the firmware is updated.

## LISTOPENS Command

LISTOPENS is a nonsensitive command that displays the number of openers for the specified object.

### LISTOPENS Command

The LISTOPENS LIF command displays OPENER information and the number of openers for a LIF.

### Command Syntax

```
LISTOPENS LIF [/OUT file-spec / ] LIF [ lif-name ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*lif-name*

is the name of the LIF object.

## LISTOPENS LIF Display

The display for the LISTOPENS LIF command is:

```

SLSA Listopens LIF \SYS.$ZZLAN.LANX

Openers  PPID      BPID      Format      Filter
$ZTCP0   ( 0,280) ( 1,301) Ethernet   $ZTCP0.#SN1.BR0
$ZTCP0   ( 0,280) ( 1,301) Ethernet   $ZTCP0.#SN1.IP0
$ZTCP0   ( 0,280) ( 1,301) Ethernet   $ZTCP0.#SN1.AR0

Total Number of Openers: 3

```

Openers

is the process name of the opener of the LIF.

PPID

is the primary processor and process ID of the opener.

BPID

is the backup processor and process ID of the opener.

Format

indicates the type of network interface.

Filter

is the name of the filter.

## NAMES Command

NAMES is a nonsensitive command that displays a list of subordinate object types and the names for the specified object.

### NAMES null Command

The NAMES null command displays the names of all processes and objects in the SLSA subsystem. The null object is not specified in the command.

### Command Syntax

```
NAMES [/OUT file-spec / ][ object-name ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*object-name*

is the name of the LAN Manager (LANMAN) process, \$ZZLAN. Omitting the *object-name* invokes the null object.

## **NAMES null Display (Issued to the SLSA Manager Process)**

The display for the NAMES null command when issued to the SLSA manager process is:

```
SLSA Names PROCESS \SYS.  
  
PROCESS  
$ZZLAN  
  
LIF  
$ZZLAN.LANX  $ZZLAN.LANY  $ZZLAN.L018  $ZZLAN.L019  $ZZLAN.L01A  
$ZZLAN.L01B  
$ZZLAN.L01C  $ZZLAN.L01D  $ZZLAN.L01E  $ZZLAN.L01F  
  
ADAPTER  
$ZZLAN.M0IE0  
  
SAC  
$ZZLAN.M0IE0.0  
  
PIF  
$ZZLAN.M0IE0.0.A
```

(Names null display continued)

```

SLSA Names PROCESS \SYS.

PROCESS
$ZZLAN

LIF
$ZZLAN.LANX  $ZZLAN.LANY  $ZZLAN.L018  $ZZLAN.L019  $ZZLAN.L01A
$ZZLAN.L01B
$ZZLAN.L01C  $ZZLAN.L01D  $ZZLAN.L01E  $ZZLAN.L01F

ADAPTER
$ZZLAN.M0IE0

SAC
$ZZLAN.M0IE0.0

PIF
$ZZLAN.M0IE0.0.A
ADAPTER
$ZZLAN.M0IE1

SAC
$ZZLAN.M0IE1.0

PIF
$ZZLAN.M0IE1.0.A

ADAPTER
$ZZLAN.E0153

SAC
$ZZLAN.E0153.0

PIF
$ZZLAN.E0153.0.A  $ZZLAN.E0153.0.B

SAC
$ZZLAN.E0153.1

PIF
$ZZLAN.E0153.1.A  $ZZLAN.E0153.1.B
ADAPTER
$ZZLAN.E0154

SAC
$ZZLAN.E0154.0

PIF
$ZZLAN.E0154.0.A  $ZZLAN.E0154.0.B

SAC
$ZZLAN.E0154.1

PIF
$ZZLAN.E0154.1.A  $ZZLAN.E0154.1.B

MON
$ZZLAN.#ZLM00  $ZZLAN.#ZLM01  $ZZLAN.#ZLM02  $ZZLAN.#ZLM03  $ZZLAN.#ZLM04
$ZZLAN.#ZLM05  $ZZLAN.#ZLM06  $ZZLAN.#ZLM07  $ZZLAN.#ZLM08  $ZZLAN.#ZLM09
$ZZLAN.#ZLM10  $ZZLAN.#ZLM11  $ZZLAN.#ZLM12  $ZZLAN.#ZLM13  $ZZLAN.#ZLM14
$ZZLAN.#ZLM15

```

## NAMES ADAPTER Command

The NAMES ADAPTER command displays the names of the SACs and PIFs subordinate to the specified adapter.

### Command Syntax

```
NAMES [ /OUT file-spec / ] ADAPTER adapter-name
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ADAPTER *adapter-name*

specifies the adapter for which a list of subordinate SAC and PIF objects is to be displayed. The adapter name has the form \$ZZLAN.*adapter-name*, for example, \$ZZLAN.G4SA0.

### NAMES ADAPTER Display (Issued to the SLSA Manager Process)

```
1-> NAMES ADAPTER $ZZLAN.mioe0

SLSA Names ADAPTER \SYS.$ZZLAN.mioe0

ADAPTER
$ZZLAN.MIOE0
```

```
2->names adapter*

SLSA Names ADAPTER \SYS.$ZZLAN.*

ADAPTER
$ZZLAN.ATM2 $ZZLAN.ATM3 $ZZLAN.E4SA0 $ZZLAN.MIOE0 $ZZLAN.MIOE1
```

```
3-> names ADAPTER $ZZLAN.a*

SLSA Names ADAPTER \SYS.$ZZLAN.a*

ADAPTER
$ZZLAN.ATM2 $ZZLAN.ATM3
```

## NAMES ATMSAP Command

The NAMES ATMSAP command displays the names of the ATMSAP objects configured.

## Command Syntax

`NAMES [ /OUT file-spec/ ] ATMSAP atmsap-name`

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*ATMSAP atmsap-name*

is the name of the target ATMSAP object. *atmsap-name* is in the form

*\$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id*

where:

*adapter-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that identifies the parent SLSA ADAPTER. Wildcard characters are allowed.

*sac-id*

is a number that identifies the parent SAC object. The only valid *sac-id* for the ATM3SA adapter is the number “0.” Wildcard characters are allowed.

*pif-id*

is an alpha character that identifies the parent PIF object. The only valid *pif-id* for the ATM3SA adapter is the letter “A.” Wildcard characters are allowed.

*atmsap-id*

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP object. Wildcard characters are allowed.

## Response Display (Issued to the SLSA Manager Process)

```
6-> names atmsap $ZZLAN.ATM2.0.A.ASAP101

SLSA Names ATMSAP \SYS.$ZZLAN.ATM2.0.A.ASAP101

ATMSAP
$ZZLAN.ATM2.0.A.ASAP101
```

```
7-> names atmsap*

SLSA Names ATMSAP \SYS.$ZZLAN.*

ATMSAP
$ZZLAN.ATM2.0.A.ASAP1 $ZZLAN.ATM2.0.A.ASAP10
$ZZLAN.ATM2.0.A.ASAP100 $ZZLAN.ATM2.0.A.ASAP101
$ZZLAN.ATM2.0.A.ASAP102 $ZZLAN.ATM2.0.A.ASAP103
```

## Example

```
NAMES ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01
```

## NAMES LIF Command

The NAMES LIF command displays the names of LIFs in the SLSA subsystem.

## Command Syntax

```
NAMES [ /OUT file-spec / ] LIF lif-name
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

LIF *lif-name*

specifies the LIF for which a name is displayed. The LIF name has the form \$ZZLAN.*lif-name*, for example, \$ZZLAN.L11123A.

## NAMES LIF Display (Issued to the SLSA Manager Process)

```
-> SCF NAMES LIF $ZZLAN.*
```

```
SLSA Names LIF \SYS.$ZZLAN
```

```
LIF
```

```
$ZZLAN.L11123A  $ZZLAN.L11123B  $L11123C  $L11123D
```

```
-> SCF NAMES LIF $ZZLAN.L11123A
```

```
SLSA Names LIF \SYS.$ZZLAN.L11123A
```

```
LIF
```

```
$ZZLAN.L11123A
```



## NAMES MON Command

The NAMES MON command displays the names of LAN Monitor (LANMON) processes.

### Command Syntax

```
NAMES [ /OUT file-spec / ] MON lanmon-name
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

MON *lanmon-name*

specifies the monitor process for which a list of subordinate monitor objects is to be displayed. The monitor name has the form \$ZZLAN.ZLM*nn*, for example, \$ZZLAN.ZLM00, where *nn* specifies the processor in which the LANMON process is running.

### NAMES MON Display

The display for the NAMES MON command when issued to the SLSA manager process is:

```
-> NAMES MON $ZZLAN.*

SLSA Names MON \SYS.$ZZLAN.$ZLM00

MON
$ZZLAN.#ZLM00 $ZZLAN.#ZLM01 $ZZLAN.#ZLM02 $ZZLAN.#ZLM03 $ZZLAN.#ZLM04
$ZZLAN.#ZLM05 $ZZLAN.#ZLM06 $ZZLAN.#ZLM07 $ZZLAN.#ZLM08 $ZZLAN.#ZLM09
$ZZLAN.#ZLM10 $ZZLAN.#ZLM11 $ZZLAN.#ZLM12 $ZZLAN.#ZLM13 $ZZLAN.#ZLM14
$ZZLAN.#ZLM15 $ZZLAN.#ZLM16
```

## NAMES PIF Command

The NAMES PIF command displays the names of the PIFs in the SLSA subsystem.

### Command Syntax

```
NAMES [ /OUT file-spec / ] PIF pif-name
      [ , SUB { ONLY | ALL | NONE } ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PIF *pif-name*

specifies the PIF for which a list of subordinate objects is to be displayed. The PIF name has the form *\$ZZLAN.adapter-name.sac-unit#.pif-unit-letter*, for example, *\$ZZLAN.G11123.0.A*.

SUB { ONLY | ALL | NONE }

controls the set of objects and subordinate objects that the command targets:

ONLY

specifies that only the names of the subordinate object are returned.

ALL

specifies that names for the named object/subordinate objects are returned. This is the default used if the SUB keyword is used but no option is selected.

NONE

specifies that only the names of the PIF objects be returned. If wildcard characters are used the names of several PIF objects can be returned. This is the default selected if the SUB keyword is not used.

## NAMES PIF Display (Issued to the SLSA Manager Process)

```
-> NAMES PIF $ZZLAN.*

SLSA Names PIF \SYS.$ZZLAN.*

PIF
$ZZLAN.M0IE0.0.A  $ZZLAN.M0IE1.0.A  $ZZLAN.G11123.0.A  $ZZLAN.G11123.0.B
$ZZLAN.G11123.0.C  $ZZLAN.G11123.0.D
```

## NAMES PROCESS Command

The NAMES PROCESS command displays the names of objects subordinate to the specified process.

### Command Syntax

```
NAMES [ /OUT file-spec / ] PROCESS $process-name
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PROCESS *\$process-name*

specifies the process for which a list of subordinate objects is to be displayed. For the LAN Manager (LANMAN) process, *\$process* is *\$ZZLAN*.

## NAMES PROCESS Display (Issued to the SLSA Manager Process)

```
-> NAMES PROCESS $ZZLAN
```

```
SLSA Names PROCESS \SYS.$ZZLAN
```

```
PROCESS  
$ZZLAN
```

```
LIF  
$ZZLAN.LANX $ZZLAN.LANY $ZZLAN.E018 $ZZLAN.L019 $ZZLAN.L01A  
$ZZLAN.L01B $ZZLAN.L01C $ZZLAN.L01D $ZZLAN.L01E $ZZLAN.L01F
```

```
ADAPTER  
$ZZLAN.MI0150
```

```
SAC  
$ZZLAN.M0IE0.0
```

```
PIF  
$ZZLAN.M0IE0.0.A
```

```
ADAPTER  
$ZZLAN.M0IE1
```

```
SAC  
$ZZLAN.M0IE1.0
```

```
PIF  
$ZZLAN.M0IE1.0.A
```

```
ADAPTER  
$ZZLAN.E0153
```

```
SAC  
$ZZLAN.E0153.0
```

## (NAMES Process Display Continued.)

```

PIF
$ZZLAN.E0153.0.A  $ZZLAN.E0153.0.B

SAC
$ZZLAN.E0153.1

PIF
$ZZLAN.E0153.1.A  $ZZLAN.E0153.1.B

ADAPTER
$ZZLAN.E0154

SAC
$ZZLAN.E0154.0

PIF
$ZZLAN.E0154.0.A  $ZZLAN.E0154.0.A

SAC
$ZZLAN.E0154.1

PIF
$ZZLAN.E0154.1.A  $ZZLAN.E0154.1.B

MON
$ZZLAN.#ZLM00  $ZZLAN.#ZLM01  $ZZLAN.#ZLM02  $ZZLAN.#ZLM03
$ZZLAN.#ZLM04
$ZZLAN.#ZLM05  $ZZLAN.#ZLM06  $ZZLAN.#ZLM07  $ZZLAN.#ZLM08
$ZZLAN.#ZLM09
$ZZLAN.#ZLM10  $ZZLAN.#ZLM11  $ZZLAN.#ZLM12  $ZZLAN.#ZLM13
$ZZLAN.#ZLM14
$ZZLAN.#ZLM15

```

## PROCESS

shows the name of the LANMAN process.

## LIF

shows the names of the logical interfaces.

## ADAPTER

shows the names of the adapters.

## SAC

shows the names of the ServerNet addressable controllers.

## PIF

shows the names of the physical interfaces.

## MON

shows the names of the LANMON processes running in each processor.

## NAMES SAC Command

The NAMES SAC command displays the names of SACs in the SLSA subsystem.

### Command Syntax

```
NAMES [ /OUT file-spec / ] SAC sac-name
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*SAC sac-name*

specifies the SAC for which a list of subordinate objects is to be displayed. The SAC name has the form *\$ZZLAN.adapter-name.sac-unit#*, for example, *\$ZZLAN.G11123.0*.

### NAMES SAC Display

```
-> NAMES SAC $ZZLAN.G11123.0

SLSA Names SAC \SYS.$ZZLAN.G11123.0

SAC
$ZZLAN.G11123.0
```

*SAC*

shows the name of the ServerNet addressable controller.

*PIF*

shows the name of the physical interfaces subordinate to the SAC.

## RESET Command

RESET is a sensitive command that moves an object to a state from which it can be started. This command interrupts a pending operation or takes an object out of a state.

The RESET ADAPTER command is intended mainly for resetting the POST-fail LED so that the LED reflects the current result of a POST. Do not use the RESET ADAPTER command during normal operation.

## RESET ADAPTER Command

The RESET ADAPTER command initiates a hardware reset of the adapter, putting it in a known state. The ADAPTER object must be in the STOPPED state before you issue this command.

### Command Syntax

```
RESET [ /OUT file-spec / ] ADAPTER adapter-name
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*ADAPTER adapter-name*

specifies which adapter to reset.

### Considerations

- This command initiates the hardware-reset operation.
- The hardware reset turns off the LED that indicates a failure of the power-on self-test (POST). The POST-fail LED turns on again only when a POST failure is detected.
- The command is not meaningful for the MFIOB.

### Example

```
RESET ADAPTER $ZZLAN.E0153
```

## START Command

START is a sensitive command that initiates the operation of the specified object. When the SLSA subsystem has successfully completed processing this command, the object is placed in the STARTED or STARTING state.

### START ADAPTER Command

The START ADAPTER command initiates the operation of the ADAPTER object. When the subsystem has successfully completed processing this command, the ADAPTER object is placed in the STARTED state.

## Command Syntax

```
START [ /OUT file-spec / ] ADAPTER adapter-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ADAPTER *adapter-name*

specifies the adapter to start.

SUB [ ONLY | ALL | NONE ]

directs the command at a set of subordinate objects.

ONLY specifies that only the subordinate objects are affected.

ALL specifies that the named object and the subordinate objects are affected.

NONE specifies that none of the subordinate objects are affected.

## Considerations

- You must start the ADAPTER object by using the START ADAPTER command after adding the adapter to the system.
- Omitting the SUB option has the same affect as SUB NONE.
- Including the SUB option without specifying ALL, NONE, or ONLY has the same affect as SUB ALL.

## Example

The following example starts the adapter named G1123:

```
START ADAPTER $ZZLAN.G1123
```

## START ATMSAP Command

The START ATMSAP command causes target ATMSAP objects to begin operation. The ATMSAP objects enter the STARTED summary state if the command is successful. The command is rejected if the ATMSAP object is already in the STARTED summary state.

```
START [ /OUT file-spec/ ] ATMSAP atmsap-name
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ATMSAP *atmsap-name*

is the name of the target ATMSAP object. *atmsap-name* is in the form

*\$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id*

where:

*adapter-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that identifies the parent SLSA ADAPTER. Wildcard characters are allowed.

*sac-id*

is a number that identifies the parent SAC object. The only valid *sac-id* for the ATM3SA adapter is the number "0." Wildcard characters are allowed.

*pif-id*

is an alpha character that identifies the parent PIF object. The only valid *pif-id* for the ATM3SA adapter is the letter "A." Wildcard characters are allowed.

*atmsap-id*

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP object. Wildcard characters are allowed.

## Response Display

The START ATMSAP command returns only a success or failure indication. Successful completion is indicated when SCF displays the prompt for the next command. A failure is indicated when SCF displays an error message.

## Example

```
START ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01
```



## START LIF Command

The START LIF command initiates the operation of the line interface (LIF) object. When the subsystem has completed processing this command, the LIF object is placed in the STARTED state.

### Command Syntax

```
START [ /OUT file-spec / ] LIF lif-name
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*LIF lif-name*

specifies the LIF to be placed in the STARTED state. The LIF name has the form \$ZZLAN.*lif-name*, for example, \$ZZLAN.L11123C.

### Considerations

- You must start the LIF object by using the START LIF command after you add the LIF object to the system.
- The START LIF command can cause the PIF associated with the specified LIF to briefly go into the STARTING state.
- The START LIF command does not affect the access state of the LIF.

### Example

The following example starts the LIF named L11123C:

```
START LIF $ZZLAN.L11123C
```

## START MON Command

The START MON command initiates the operation of a LANMON process.

### Command Syntax

```
START [ /OUT file-spec / ] MON lanmon-name
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

MON *lanmon-name*

specifies the LANMON process to be started. LANMON names have the form \$ZZLAN.#ZLM *nn* where *nn* indicates the particular processor in which the LANMON should start.

## Example

The following example starts the LANMON named ZLM01 in processor 01:

```
START MON $ZZLAN.ZLM01
```

## START PIF Command

The START PIF command initiates the operation of the physical interface (PIF) object. When the subsystem has completed processing this command, the PIF object is placed in the STARTED (or STARTING) state.

## Command Syntax

```
START [ /OUT file-spec/ ] PIF pif-name
      [ , SUB { ONLY | ALL | NONE } ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PIF *pif-name*

is the name of the PIF to be started. The PIF name has the form *adapter-name.sac-unit#.pif-unit-letter*, for example, G11123.0.A.

SUB { ONLY | ALL | NONE }

controls the set of objects and subordinate objects that the command targets.

The following values are valid:

ONLY

specifies that only the subordinate objects are targets of the command.

ALL

specifies that the named object and the subordinate objects are targets of the command. This is the default used if the SUB keyword is used, but no option is selected.

NONE

specifies that none of the subordinate objects are targets of the command. This is the default selected if the SUB keyword of the SUB keyword is not used.

## Considerations

- The parent SAC object must be in the STARTED or STARTING state for the START PIF command to be accepted. The PIF object goes into the STARTED state only when the parent SAC object is in the STARTED state.
- The command returns no error if the PIF object goes to the STARTING state. To see errors that prevent the PIF object from going to the STARTED state, use EMS.
- The PIF cannot go to the STARTED state until the ServerNet adapter is successfully downloaded with the DLFILENAME (see [INFO SAC Command](#) on page 4-60). START SAC causes the download of the SAC. A STATUS SAC of STARTED means the SAC has been successfully downloaded.
- The PIF on a TRSA cannot go to the STARTED state until successful ring insertion occurs. When not connected into a token-ring LAN, the PIF remains in a STARTING state and SLSA event number 4101 is issued.
- TRSA PIF STARTED state signifies a successful download to the adapter and ring insertion.
- After starting the PIF, issue the INFO SAC, DETAIL command to check the configuration of the SAC. If there is a DUMP file (see [DumpFilename](#) on page 4-62) in the SYSnn, check its creation date. Provide the dump file to your support representative if the cause of the failure is unknown. Also, provide any related EMS messages for the SLSA subsystem and VPROC of the DLFILENAME.

## Example

The following is an example of the START PIF command:

```
START PIF G11123.0.A
```

## START SAC Command

The START SAC command initiates the specified SAC and places it in the STARTED state.

## Command Syntax

```
START [ /OUT file-spec/ ] SAC sac-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

SAC *sac-name*

is the name of the SAC to be started. The SAC name has the form *\$ZZLAN.adapter-name.sac unit#*, for example, *\$ZZLAN.G11123.0*.

SUB [ ONLY | ALL | NONE ]

directs the command at a set of subordinate objects.

ONLY specifies that only the subordinate objects are affected.

ALL specifies that the named object and the subordinate objects are affected.

NONE specifies that none of the subordinate objects are affected.

## Considerations

- The PIF objects subordinate to the SAC are not started unless you specify the SUB ALL or SUB ONLY options with the command; otherwise, you must issue START PIF commands for the PIF objects.
- The command returns no error if the SAC object goes to the STARTING state. Errors that prevent the SAC object from going to the STARTED state can be seen through the Event Management Service (EMS).
- Omitting the SUB option has the same affect as SUB NONE.
- Including the SUB option without specifying ALL, NONE, or ONLY has the same effect as SUB ALL.

## Example

The following is an example of the START SAC command:

```
START SAC $ZZLAN.G11123.0
```

# STATISTICS Command

The STATISTICS command displays statistical information for the specified object.

## STATS ATMSAP Command

The STATS ATMSAP command returns the usage information for an ATMSAP object subordinate to a PIF object.

### Command Syntax

```
STATS [ /OUT file-spec/ ATMSAP atmsap-name [, RESET ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ATMSAP *atmsap-name*

is the name of the target ATMSAP object. *atmsap-name* is in the form

*\$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id*

where:

*adapter-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that identifies the parent SLSA ADAPTER. Wildcard characters are allowed.

*sac-id*

is a number that identifies the parent SAC object. The only valid *sac-id* for the ATM3SA adapter is the number "0." Wildcard characters are allowed.

*pif-id*

is an alpha character that identifies the parent PIF object. The only valid *pif-id* for the ATM3SA adapter is the letter "A." Wildcard characters are allowed.

*atmsap-id*

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP object. Wildcard characters are allowed.

RESET

causes the statistical counters for the target objects to be set to zero. If RESET is selected, the STATS command is treated as a sensitive command.

## Response Display

The STATS ATMSAP command displays the usage statistics for ATMSAP objects. The display format is shown below. A failure is indicated when SCF displays an error message.

```
3-> stats ATMSAP $zzlan.atm1.0.a.atmsap01

STATS ATMSAP \SYS.$ZZLAN.ATM1.0.A.ATMSAP01

Sample Time           26 August 1999, 17:18:04.643
Reset Time            26 August 1999, 17:07:03.021

DATA Pkts Tx          0D          Data Pkts Rx          0D
Discarded Pkts Tx     0D          Discarded Pkts Rx     0D
```

Sample Time

is the time when the statistics were sampled.

Reset Time

is the time when the statistical counters were last reset.

Data Pkts Tx or Rx

is the total user data packets transmitted or received.

Discarded Pkts Tx

is the total user data packets discarded.

## Considerations

- The statistical counters are reset by the subsystem when an ABORT or STOP command is processed.
- The STATS ATMSAP command should be issued only when the object is in the started summary state.

## Example

```
STATS ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01
```

## STATS PIF Command

The STATS PIF command displays statistical information for the specified PIF. This is a nonsensitive command except when the RESET option is specified.

## Command Syntax

```
STATS [ /OUT file-spec / ] PIF pif-name [ , RESET ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PIF *pif-name*

is the name of the PIF for which statistics are to be displayed. The PIF name has the form *adapter-name.sac-unit#.pif-unit-letter*, for example, G11123.0.A.

RESET

restores all the statistics counters to their initial values after the statistics are displayed. This is a sensitive option.

## Considerations

- Statistics are not provided for PIF objects that are part of a MFIOB.
- Statistics are accessible only when the specified PIF is in the STARTED state.
- PIF statistics are reset when a connection is established between the host and the adapter; that is, when the PIF changes from the STARTING state to the STARTED state. The PIF statistics also are reset whenever ownership of a SAC changes.

## STATS PIF Display

Displays are slightly different for most adapters.

### STATS PIF Display for an ATM3SA Adapter

```
-> stats pif $zzlan.a0154.0.A
SLSA Stats PIF \SYS.$ZZLAN.A0154.0.A

Sample Time.....31 Jul 2001, 12:45:36.155
Reset Time.....30 Jul 2001, 14:26:05.772

DFC Events.....60587 DFTEvents.....7614

Exec Rate (per sec).....52575 Exec Rate LWM.....45462
Service Time (usec).....42    Service Time HWM.....32516

TNet Halt 0.....0          TNet Halt 1.....0
TNet Link Exception 0....0    TNet Link Exception 1     0
TNet Packet Exception 0..0    TNet Packet Exception 1..0
TNet Timeout.....0          TNet Nack.....0

Cells Received.....15389 Cells Transmitted.....10569
Cells Dropped.....0         Frames Dropped.....0
Congestion.....0           Free Buffer Underflow....0
Fifo Overrun.....0          Max Length Error.....0
CRC Error.....0             User Abort.....0
Length Error.....0          Tl Abort.....0
Channel Deactivate.....0     Raw CRC Error.....0
```

#### Sample Time

indicates the time when the statistics were taken.

#### Reset Time

indicates when the statistics counters were last reset.

#### DFC Events

indicates the number of times data has been forwarded based on Data Forward Count being reached.

#### DFT Events

indicates the number of times data has been forwarded based on Data Forward Time expiration.

#### Exec Rate

indicates the number of times the adapter has completed processing the main execution loop in the last second. This can be used as a guide to determine the relative processor loading of the adapter.

#### Exec Rate HWM

High water mark for Exec Rate.



### Service Time

indicates the number of microseconds spent to complete processing of the last iteration of the main processing loop. This is an indication of loading similar to the exec rate.

### Service Time HWM

indicates the longest time (in microseconds) spent in a single iteration of the main processing loop.

### TNet Halt 0

indicates the number of TNet Halt 0 exceptions.

### TNet Halt 1

indicates the number of TNet Halt 1 exceptions.

### TNet Link Exception 0

indicates the number of TNet Link 0 exceptions.

### TNet Link Exception 1

indicates the number of TNet Link 1 exceptions.

### TNet Packet Exception 0

indicates the number of TNet Packet 0 exceptions.

### TNet Packet Exception 1

indicates the number of TNet Packet 1 exceptions.

### TNet Timeout

indicates the number of TNet Time-outs received. All of the TNet related statistics can help diagnose ServerNet problems or congestion.

### TNet Nack

indicates the number of TNet Nacks received.

**ATM3SA Line Stats****Cells Received**

counts the number of cells received on the specified line.

**Cells Transmitted**

counts the number of cells transmitted on the specified line.

**Cells Dropped**

counts the number of cells dropped on the specified line.

**Frames Dropped**

counts the number of cells dropped on the specified line.

**Congestion**

counts the amount of congestion that has been experienced on the network.

**Free Buffer Underflow**

counts the number of free buffer underflow errors that have occurred while NEC was receiving packets.

**Fifo Overrun**

counts the number of FIFO overrun errors that have occurred while NEC was receiving packets.

**Max Length Error**

indicates the number of received cells that exceed the maximum number of segments shown in the VC table.

**CRC Error**

counts the number of CRC errors that occurred while NEC was receiving AAL-5 packets.

**User Abort**

counts the number of times the length of the received packet did not match the packet size.

**Length Error**

counts the number of times the length of the received packet did not match the packet size.

**T1 Error**

counts the number of times the packet did not arrive within the T1 time limit.

**Channel Deactivate**

counts the number of times the channel was deactivated by the REACTIVATED\_CHANNEL command.

**Raw CRC Error**

counts the number of times an error occurred while NEC was receiving a non-AAL-5 packet.

**STATS PIF Display for an E4SA or FESA Adapter**

```
-> STATS PIF $ZZLAN.E0153.0.A

SLSA Stats PIF \SYS.$ZZLAN.E0153.0.A

Sample Time..... 30 March 2000, 10:41:56.723
Reset Time..... 30 March 2000, 10:40:54.197

Octets Transmitted OK..... 1050412
Octets Received OK..... 3035904

      General Send Stats                General Receive Stats
Unicast Frames Transmitted.. 10722      Unicast Frames Received.... 10730
Non-Unicast Frames Transmit. 9          Non-Unicast Frames Received 8775
Frames Discarded..... 0              Frames Discarded..... 0
Errors..... 0                        Errors..... 0
Current OutQueue Length .... 0          Unknown Protocol Frames.... 0

      802.3 Send Stats                  802.3 Receive Stats
Transmit Underruns..... 0              Receive Overruns..... 0
Defer Packets..... 0                  CRC Errors..... 0
Send Late Collison..... 0             Receive Late Collision..... 0
Total Collision Frames..... 0          Alignment Errors..... 0
Excessive Collisions..... 0           Receive Long Frame..... 0
Total Collisions..... 0               Receive Short Frame..... 0
Heartbeat Check Errors..... 0
Loss of Carrier Sense..... 0
```

**Time Stats****Sample Time**

indicates the time when the statistics were taken.

**Reset Time**

indicates when the statistics counters were last reset.

**Octet Stats****Octets Transmitted OK**

indicates the number of bytes transmitted to the LAN.

Octets Received OK

indicates the number of bytes received from the LAN.

## General Send Stats

Unicast Frames Transmitted

indicates the number of unicast frames transmitted to the LAN.

Non-Unicast Frames Transmit

indicates the number of non-unicast frames transmitted to the LAN.

Frames Discarded

indicates the number of frames discarded due to resource problems or Ethernet errors.

Errors

indicates the number of frames discarded due to errors.

Current OutQueue Length

indicates the length of the QIO queue for out-bound frames when the statistics sample was taken.

## General Receive Stats

Unicast Frames Received

indicates the number of unicast frames received from the LAN.

Non-Unicast Frames Received

indicates the number of non-unicast frames received from the LAN.

Frames Discarded

indicates the number of frames discarded due to resource problems or Ethernet errors.

Errors

indicates the number of errors that occurred during reception.

Unknown Protocol Frames

indicates the number of frames discarded because the filter could not identify a destination.

## 802.3 Send Stats

### Transmit Underruns

indicates the number of frames that had their transmission aborted because memory could not be accessed quickly enough to allow transmission.

### Defer Packets

indicates the number of frames that could not be transmitted immediately because the LAN was in use.

### Send Late Collision

indicates the number of frames that had their transmission aborted because a collision occurred before the completion of the retry process.

### Total Collision Frames

indicates the number of frames that had at least one collision during transmission.

### Excessive Collisions

indicates the number of frames that had their transmission aborted after having 15 collisions.

### Total Collisions

indicates the total number of collisions detected during transmission of frames to the LAN.

### Heartbeat Check Errors

indicates the number of times a heartbeat error occurred.

### Loss of Carrier Sense

indicates the number of frames that had their transmission aborted because carrier sense was lost.

## 802.3 Receive Stats

### Receive Overruns

indicates the number of frames discarded because memory could not be accessed fast enough to allow reception.

### CRC Errors

indicates the number of frames discarded because of frame checksum errors.

**Receive Late Collision**

indicates the number of frames discarded because a collision occurred before the completion of the retry process.

**Alignment Errors**

indicates the number of frames discarded because the frame did not end on a byte boundary.

**Receive Long Frame**

indicates the number of frames discarded because the length of the frame was greater than the maximum frame size.

**Receive Short Frame**

indicates the number of frames discarded because the length of the frame was smaller than the minimum frame size.

**STATS PIF Display for a TRSA**

```
-> STATS PIF $ZZLAN.T0154.0.A

SLSA Stats PIF \SYS.$ZZLAN.T0154.0.A

Sample Time..... 30 March 2000, 15:06:02.146
Reset Time..... 30 March 2000, 12:03:03.279

Octets Transmitted OK..... 1050412
Octets Received OK..... 3035904

      General Send Stats                      General Receive Stats
Unicast Frames Transmitted.. 10722          Unicast Frames Received.... 10730
Non-Unicast Frames Transmit. 9              Non-Unicast Frames Received 8775
Frames Discarded..... 0                  Frames Discarded..... 0
Errors..... 0                            Errors..... 0
Current OutQueue Length ... 0              Unknown Protocols Frames... 0

      802.5 (Token Ring) Stats
Line Errors..... 0                      Burst Errors..... 0
AC Errors..... 1                      Abort Transmit Errors..... 0
Internal Errors..... 0                  Lost Frame Errors..... 0
Receive Congestions..... 0              Frame Copied Errors..... 0
Token Errors..... 0                    Soft Errors..... 0
Hard Errors..... 0                     Signal Loss Errors..... 0
Transmit Beacons..... 0                 Recoveries..... 0
Lobe Wires..... 0                      Removes..... 0
Singles..... 0                         Frequency Errors..... 0
```

**Time Stats****Sample Time**

indicates the time when the statistics were taken.

**Reset Time**

indicates when the statistics counters were last reset.

## Octet Stats

Octets Transmitted OK

indicates the number of bytes successfully transmitted to the LAN.

Octets Received OK

indicates the number of bytes successfully received from the LAN.

## General Send Stats

Unicast Frames Transmitted

indicates the number of unicast frames transmitted to the LAN.

Non-Unicast Frames Transmit

indicates the number of non-unicast frames transmitted to the LAN.

Frames Discarded

indicates the number of frames discarded due to resource problems or Ethernet errors.

Errors

indicates the number of frames discarded due to errors.

Current OutQueue Length

indicates the length of the QIO queue for out-bound frames when the statistics sample was taken.

## General Receive Stats

Unicast Frames Received

indicates the number of unicast frames successfully received from the LAN.

Non-Unicast Frames Received

indicates the number of non-unicast frames successfully received from the LAN.

Frames Discarded

indicates the number of frames discarded due to resource problems or Ethernet errors.

Errors

indicates the number of errors that occurred during reception.

### Unknown Protocol Frames

indicates the number of frames discarded because the filter could not identify a destination.

## 802.5 Stats

### Line Errors

indicates that this adapter repeated or copied a frame or token and a violation of the hardware-clocking protocol was detected. A high count in this field might indicate a hardware problem with the upstream neighborhood of this adapter.

### Burst Errors

indicates that this adapter repeated or copied a frame or token and a violation of the hardware clocking protocol was detected. A high count in this field might indicate a hardware problem in the upstream neighborhood of this adapter.

### AC Errors

indicates that the upstream neighbor of this adapter could not set the ARI/FCI bits in a Monitor Present frame. A high count in this field might indicate a hardware problem in the upstream neighborhood of this adapter.

### Abort Transmit Errors

indicates this adapter experienced a problem with the DMA circuit. A high count in this field may indicate a hardware problem in the adapter.

### Internal Errors

indicates an unexpected error has occurred in the adapter. A non-zero count in this field may indicate a hardware problem in the adapter.

### Lost Frame Errors

this counter indicates that this adapter was attempting to strip a frame transmitted from the ring and was unable to receive the end-of-frame. A high count in this field might indicate a hardware problem in the upstream neighborhood of this adapter.

### Receive Congestions

is incremented each time a frame addressed to this adapter cannot be copied from the ring due to lack of local buffer space on the adapter. A high count in this field may indicate this adapter is supporting too many connections and needs a higher transfer rate.

### Frame Copied Errors

is incremented when this adapter recognizes a frame addressed to its specific address but finds that the ARI bits are not set to 0. A high count in this field



indicates line hits, an adapter with a duplicate address, or a malfunctioning adapter in the local ring.

#### Token Errors

is incremented only when this adapter is the active monitor and detects an error in the token protocol. High counts in this field might indicate a hardware problem in another adapter in the local ring.

#### Soft Errors

is incremented when this adapter transmits a soft-error MAC frame. A soft error is a transient error which does not affect normal ring operation, and can be corrected by the higher-layer communication protocols. A high count in this field might indicate a hardware or noise problem in the local ring.

#### Hard Errors

is incremented when this adapter is presently transmitting and receiving Beacon MAC frames. A high count in this field indicates an error condition requiring operator investigation.

#### Signal Loss Errors

indicates that this adapter has detected a temporary loss of signal on the local ring. A high count in this field may indicate a malfunction somewhere in the local ring, and operator investigation is required.

#### Transmit Beacons

is incremented when this adapter transmits beacons to the local ring. This condition may indicate that a station tried to enter the ring at the wrong speed or a transient error was detected. A high count in this field might indicate that operator investigation is needed.

#### Recoveries

is incremented when this adapter detects claim-token MAC frames on the ring. A high count in this field might indicate that the ring cannot recover automatically from an error, and that operator investigation is needed.

#### Lobe Wires

is incremented when the adapter detects an open or shorted circuit between the adapter and the multistation access unit (MAU). A non-zero count indicates that operator investigation is required.

#### Removes

is incremented when this adapter has received a remove-ring-station MAC frame. An incrementing count in this field indicates a hardware fault in the local adapter.

Singles

is incremented each time this adapter recognizes that it is the only adapter on the ring. This field is provided for informational purposes only.

Frequency Errors

is incremented when this adapter detects that its incoming frames are being transmitted at the wrong frequency. A non-zero count might indicate a hardware problem in the adapter or its upstream neighbor.

STATS PIF Display for a GESA

```

SLSA Stats PIF \SYS.$ZZLAN.G0153.0.A

Sample Time..... 25 Jan 2002, 10:44:41.691
Reset Time..... 24 Jan 2002, 6:57:48.854

DFC Events..... 0
DFT Events..... 0
Exec Rate..... 0
Exec Rate LWM..... 0
TNET NACKS..... 0
TNET Timeouts..... 0
In Frame Discard Errors..... 0
Out Frame Queue Length..... 0
Out Frame Discard Errors..... 0

      802.3 Send Stats
Internal MAC TX Errors..... 0
Single Collision Frames..... 0
Multi Collision Frames..... 0
Deferred Transmissions..... 0
Excess Collision..... 0
Late Collision..... 0
Carrier Sense Errors..... 0
Octets..... 0
Unicast Packets..... 0
Multicast Packets..... 0
Broadcast Packets..... 0
XON Packets Sent..... 0
XOFF packets Sent..... 0
```

(STATS PIF display for a GESA continued)

```

Packets Discards..... 0
Packets Errors..... 0
Total Collisions..... 0
Flow Control Done..... 0

      802.3 Receive Stats
FCS Errors..... 0
Alignment Errors..... 0
Frames Too Long..... 0
XON Pause Frames Received..... 0
XOFF Pause Frames Received..... 0
XOFF State Entered..... 0
Mac Control Frames Received..... 0
Octets..... 0
Unicast Packets..... 0
Multicast Packets..... 0
Broadcast Packets..... 0
In Range Length Errors..... 0
Out Range Length Errors..... 0
Packets Exceeded Jabbers Time..... 0
Undersize Packets..... 0
Fragments..... 0
Packets 64 Octets..... 0
Packets 65-127 Octets..... 0
Packets 128-255 Octets..... 0
Packets 256-511 Octets..... 0
Packets 512-1023 Octets..... 0
Packets 1024-1522 Octets..... 0
Packets 1523-2047 Octets..... 0
Packets 2048-4095 Octets..... 0
Packets 4096-8191 Octets..... 0
Packets 8192-9022 Octets..... 0

```

Sample Time

indicates the time when the statistics were taken.

Reset Time

indicates when the statistics counters were last reset.

DFC Events

data forwarded because of DFC expiration.

DFT Events

data forwarded because of DFT expiration.

Exec Rate

number of times each second the main loop has been run.

Exec Rate LWM

low water mark for execution rate.

**TNET NACKS**

number of TNET NACKS seen by the SNET hardware.

**TNET Timeouts**

number of times TNET Time-out was seen by SNET Hardware.

**In Frames Discard Errors**

number of inbound frames discarded by software because of an internal resource problem.

**Out Frame Queue Length**

outbound frame queue length when stats are sampled.

**Out Frames Discard Errors**

number of outbound frames discarded by software because of an internal resource problem.

**802.3 Send Stats****Internal MAC TX Errors**

a count of frames for which transmission on a particular interface fails due to an internal MAC sub-layer transmit error.

**Single Collision Frames**

a count of successfully transmitted frames on a particular interface for which transmission is inhibited by exactly one collision.

**Multi Collision Frames**

a count of successfully transmitted frames on a particular interface for which transmission is inhibited by more than one collision.

**Deferred Transmissions**

a count of frames for which the first transmission attempt on a particular interface is delayed because the medium is busy.

**Excess Collision**

a count of frames for which transmission on a particular interface fails due to excessive collisions.

**Late Collision**

the number of times that a collision is detected on a particular interface later than 512 bit-times into the transmission of a packet.

#### Carrier Sense Errors

the number of times that the carrier sense condition was lost or never asserted when attempting to transmit a frame on a particular interface.

#### Octets

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

#### Ucast Packets

the number of packets that the higher-level protocols requested be transmitted, and which were not addressed to a multicast or broadcast address at this sub-layer, including those that were discarded or not sent.

#### Multicast Packets

the number of packets that the higher-level protocols requested be transmitted, and which were addressed to a multicast address at this sub-layer, including those that were discarded or not sent.

#### Broadcast Packets

the number of packets that the higher-level protocols requested be transmitted, and which were addressed to a broadcast address at this sub-layer, including those that were discarded or not sent.

#### XON Packets Sent

number of XON packets sent.

#### XOFF Packets Sent

number of XOFF packets sent.

#### Packets Discards

the number of outbound packets which were chosen to be discarded even though no errors had been detected to prevent transmission of the packets.

#### Packets Errors

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

#### Total Collisions

number of total collisions.

#### Flow Control Done

flow control done.

## 802.3 Receive Stats

### FCS Errors

a count of frames received on a particular interface that are an integer number of octets in length and do not pass the FCS check.

### Alignment Errors

a count of frames received on a particular interface that are not an integer number of octets in length and do not pass the FCS check.

### Frames Too Long

a count of frames received on a particular interface that exceed the maximum permitted frame size.

### XON Pause Frames Received

MAC control frames with pause command and length equal to zero.

### XOFF Pause Frames Received

MAC control frames with pause command and length greater than zero.

### XOFF State Entered

Transmitting is disabled.

### Mac Control Frames Received

MAC control frames with no pause command.

### Octets

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

### Unicast Packets

the number of packets, delivered by this sub-layer to a higher (sub)-layer, which were not addressed to a multicast or broadcast address at this sub-layer.

### Multicast Packets

the number of packets, delivered by this sub-layer to a higher (sub)-layer, which were addressed to a multicast address at this sub-layer.

### Broadcast Packets

the number of packets, delivered by this sub-layer to a higher (sub)-layer, which were addressed to a broadcast address at this sub-layer.

## In Range Length Errors

frames with length not equal to actual bytes received.

## Out Range Length Errors

frames with type greater than 1522 and less than 1536.

## Packets Exceeded Jabbers Time

frames exceeded jabber time.

## Undersize Packets

frames with size less than 64 bytes.

## Fragments

frame size less than 64 bytes with bad FCS

## Packets 64 Octets

frame size equal to 64 bytes.

## Packets 65-127 Octets

frame size between 65 and 127 bytes, inclusive.

## Packets 128-255 Octets

frame size between 128 and 255 bytes, inclusive.

## Packets 256-511 Octets

frame size between 256 and 511 bytes, inclusive.

## Packets 512-1023 Octets

frame size between 512 and 1023 bytes, inclusive.

## Packets 1024-1522 Octets

frame size between 1024 and 1522 bytes, inclusive.

## Packets 1523-2047 Octets

frame size between 1523 and 2047 bytes, inclusive.

## Packets 2048-4095 Octets

frame size between 2048 and 4095 bytes, inclusive.

## Packets 4096-8191 Octets

frame size between 4096 and 8191 bytes, inclusive.

## Packets 8192-9022 Octets

frame size between 8192 and 9022 bytes, inclusive.

## STATS PIF Display for a G4SA

```

-> STATS PIF $ZZLAN.G11123.0.A

      SLSA Stats PIF \SYS.$ZZLAN.G11123.0.A

      Sample Time..... 29 Sep 2003, 12:34:23.098
      Reset Time..... 29 Sep 2003, 12:34:23.098

      DFC Events..... 0
      DFT Events..... 0
      Exec Rate..... 0
      Exec Rate LWM..... 0
      Out Frame Queue Length..... 0
      In Frame Discard Errors..... 0
      In Filter Frame Discards..... 0
      In Filters Added For IP Fragments..... 0
      TNET Nacks..... 0
      TNET Timeouts..... 0

      802.3 Send Stats
      Octets..... 0
      Packets..... 0
      Underrun Packets..... 0
      Single Collision Frames..... 0
      Multi Collision Frames..... 0
      Deferred Transmissions..... 0
      Excessive Collision..... 0
      Late Collision..... 0
      Signal Quality Error Fail..... 0
      Transmit Errors..... 0
      Multicast Pkts sent by higher layer.... 0
      Broadcast Pkts sent by higher layer.... 0
      Carrier Sense Errors..... 0
      Packets Discarded..... 0

      802.3 Receive Stats
      Octets..... 0
      Packets..... 0
      FCS Errors..... 0
      Alignment Errors..... 0
      Control Pause Packets..... 0
      Overrun Packets..... 0
      Bad Packets..... 0
      Runt packets..... 0
      Packet Too Short Errors..... 0
      Packet Too Long Errors..... 0
      In Range Length Errors..... 0
      Out Range Length Errors..... 0

```

### DFC Events

is the data forwarded because of DATAFORWARDCOUNT (DFC) expiration.

### DFT Events

is the data forwarded because of DATAFORWARDTIME (DFT) expiration.

### Exec Rate

is the number of times each second that the main loop has been executed.



Exec Rate LWM

is the low water mark for exec rate.

Out Frame Queue Length

is the number of outbound frames queued when the stats were sampled.

In Frame Discard Errors

is the number of outbound frames discarded by the software because of internal resource problems.

In Filter Frame Discards

is the number of frames discarded by the G4SA adapter's filter logic.

In Filters Added For IP Fragments

is the number of filters added for the IP Fragments that arrived.

TNET NACKS

is the number of TNET NACKS seen by the SNET hardware.

TNET Timeouts

is the number of times the TNET Timeout was seen by SNET Hardware.

### **802.3 Send Stats**

Octets

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

Packets

is the number of error-free packets that have been transmitted on the line.

Underrun Packets

is the number of transmit underrun packets.

Single Collision Frames

is a count of successfully transmitted frames on a particular interface for which transmission is inhibited by exactly one collision.

Transmit Errors

is the number of transmit errors when stats were sampled.

#### Multi Collision Frames

is a count of successfully transmitted frames on a particular interface for which transmission is inhibited by more than one collision.

#### Deferred Transmissions

is a count of frames for which the first transmission attempt on a particular interface is delayed because the medium is busy.

#### Excessive Collision

is a count of frames for which transmission on a particular interface fails due to excessive collisions.

#### Late Collision

is the number of times that a frame collided outside of the collision window. This is only applicable in half-duplex mode.

#### Signal Quality Error Fail

indicates that the Signal Quality Error test failed during packet transmission.

#### Multicast Pkts sent by higher layer

indicates the number of packets the higher level layer sent to a multicast address, including the frames that were discarded or not sent.

#### Broadcast Pkts sent by higher layer

is the number of packets the higher level layer sent to a broadcast address, including the frames that were discarded or not sent.

#### Carrier Sense Errors

indicates the number of times that the carrier-sense condition was lost or never asserted when attempting to transmit a frame on a particular interface.

#### Packets Discarded

is the number of outbound packets which were chosen to be discarded even though no errors have been detected to prevent their being transmitted.

### 802.3 Receive Stats

#### Octets

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

#### Packets

is the number of error-free packets that have been received on the line.

### FCS Errors

indicates a count of frames received on a particular interface that are an integral number of octets in length and do not pass the FCS check.

### Alignment Errors

indicates a count of frames received on a particular interface that are not an integral number of octets in length and do not pass the FCS check.

### Control Pause Packets

is the number of control pause packets received.

### Overflow Packets

is the number of overflow packets received.

### Bad Packets

is the number of packets with early termination caused by packet error.

### Runt packets

is the number of runt packets received.

### Packet Too Short Errors

is the number of short event packets received.

### Frames Too Long

is a count of frames received on a particular interface that exceed the maximum permitted frame size.

### In Range Length Errors

indicates the frames that have a length not equal to actual bytes received.

### Out Range Length Errors

indicates the frames that have a type greater than 1500 and less than 1536.

## STATUS Command

STATUS is a nonsensitive command that displays the current settings of an object. STATUS differs from INFO in that STATUS shows the current, dynamic settings for an object, and INFO shows the configured, static settings. Fields common to both the INFO and STATUS commands, such as line/interface speed, can differ; STATUS PIF could show line speed of 10 Mbit/sec while INFO PIF could show 100 Mbit/sec. The line speed could have changed from what was configured (shown by the INFO PIF command) because of an incompatible line speed at the hub.

## STATUS ADAPTER Command

The STATUS ADAPTER command displays the current state of the specified adapter. The state of an ADAPTER object can be either STARTED or STOPPED.

### Command Syntax

```
STATUS [ /OUT file-spec/ ] ADAPTER adapter-name [ , DETAIL ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ADAPTER *adapter-name*

is the name of the adapter for which the status is to be returned. The adapter name has the form \$ZZLAN.*adapter-name*, for example, \$ZZLAN.G11123.

DETAIL

specifies that additional status information is displayed.

### STATUS ADAPTER Display

The format of the display for the STATUS ADAPTER command is:

```
-> STATUS ADAPTER $ZZLAN.*

SLSA Status ADAPTER

Name                State
$ZZLAN.CC1          STARTED
$ZZLAN.M0IE0         STARTED
$ZZLAN.E0153         STARTED
$ZZLAN.M0IE1         STOPPED
$ZZLAN.E0154         STARTED
$ZZLAN.E0152         STARTED
```

The format of the display for the STATUS ADAPTER command with the DETAIL option is shown below for a G4SA (the display format for all adapters is similar):

```
-> STATUS ADAPTER $ZZLAN.G11123 , DETAIL

SLSA Detailed Status ADAPTER \SYS.$ZZLAN.G11123

SP Cru Presence Status... Enabled ( 5 )
SP Cru Test Status..... Test OK ( 0x0004 )
State..... STARTED
```

#### SP Cru Presence Status

indicates service processor (SP) information about the presence of a physical adapter in the location associated with the specified adapter.

#### SP Cru Test Status

indicates SP information about the result of SP tests on the physical adapter in the location associated with the specified adapter. The values are Test OK and Config OK.

#### State

indicates the current state of the adapter.

## STATUS ATMSAP Command

The STATUS ATMSAP command returns the operational-state information for an ATMSAP object subordinate to a PIF object.

### Command Syntax

```
STATUS [ /OUT file-spec/ ATMSAP atmsap-name [, DETAIL ]
```

#### OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

#### ATMSAP *atmsap-name*

is the name of the target ATMSAP object. *atmsap-name* is in the form

*\$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id*

where:

#### *adapter-id*

is a 1- to 8-character alpha-numeric string, beginning with an alpha character, that identifies the parent SLSA ADAPTER. Wildcard characters are allowed.

#### *sac-id*

is a number that identifies the parent SAC object. The only valid *sac-id* for the ATM3SA adapter is the number "0." Wildcard characters are allowed.

#### *pif-id*

is an alpha character that identifies the parent PIF object. The only valid *pif-id* for the ATM3SA adapter is the letter "A." Wildcard characters are allowed.

atmsap-id

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP object. Wildcard characters are allowed.

DETAIL

requests that detailed operational state information be displayed. The dynamically created ARP table entries subordinate to the ATMSAP object are displayed when this option is specified.

## Response Display

The display format for the STATUS ATMSAP command without the DETAIL option selected is as follows:

Status ATMSAP for \SYS.\$ZZLAN		
Name	MTU Size	State
ATM1.0.A.ATMSAP01	8192	STARTED
ATM2.0.A.ATMSAP02	8192	STARTED

The display format for the STATUS ATMSAP command with the DETAIL option selected is as follows:

DETAIL Status ATMSAP \SYS.\$ZZLAN.ATM1.0.A.ATMSAP01	
ATMSAP Summary State	STARTED
MTU Size	8192
Nature	PVC
VCC (VPI,VCI)	( 0,100 )

ATMSAP Summary State

is the SCF summary state for the ATMSAP object.

Valid values:

STARTING

STARTED

STOPPED

ATMSAP SAP ID

is the SAP identifier used to access the ATMSAP object.

MTU Size

is the MTU size currently active for the ATMSAP object.

VCC (VPI,VCI)

identifies the ARP server virtual circuit connection with its virtual channel identifier (VCI) and virtual path identifier (VPI).

VCC Nature

identifies the connection as being from a PVC or an SVC.

Remote ATM Address

ATM address for the destination.

## Example

```
STATUS ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01
```

## STATUS LIF Command

The STATUS LIF command displays the current summary state and access state of the specified LIF. The summary state of a LIF object can be either STARTED or STOPPED. The access state is either UP or DOWN.

## Command Syntax

<pre>STATUS [ /OUT <i>file-spec</i>/] LIF <i>lif-name</i> [ , DETAIL ]</pre>
--

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

LIF *lif-name*

specifies the LIF for which the status is to be displayed. The LIF name has the form \$ZZLAN.*lif-name*, for example, \$ZZLAN.L11123A.

DETAIL

specifies that additional status information is displayed.

## Considerations

More than one processor can have a data path to a LIF.

# STATUS LIF Display

The format of the display for the STATUS LIF command without the DETAIL option is:

```
-> STATUS LIF $ZZLAN.L*

SLSA Status LIF

Name                State        Access State
$ZZLAN.CC10A        STARTED        UP
$ZZLAN.L11123A      STARTED        UP
$ZZLAN.L11123B      STARTED        DOWN
$ZZLAN.L11123C      STARTED        UP
$ZZLAN.L11123D      STARTED        UP
$ZZLAN.LANX         STARTED        UP
$ZZLAN.LANY         STARTED        UP
```

## State

indicates the current state of the LIF.

## Access State

indicates whether a LIF can currently access the PIF.

The format of the display for the STATUS LIF command with the DETAIL option is:

```
-> STATUS LIF $ZZLAN.L11123A , DETAIL

SLSA Detailed Status LIF \SYS.$ZZLAN.L11123A

Access State..... UP
CPUs with Data Path..... ( 0, 1, 2 )
Potential Access CPUs.... ( 0, 1, 2, 3 )
State..... STARTED
Trace Filename.....
Trace Status.....
```

## Access State

indicates whether there is at least one processor with a data path through the PIF associated with the specified LIF. UP indicates at least one processor with a data path. DOWN indicates no data path.

## CPUs with Data Path

lists processors that have a successful data path for the PIF associated with the specified LIF. NONE means that no processor has a data path through the PIF. The processors are listed in numerical order.

## Potential Access CPUs

lists processors that can potentially have a data path through the PIF associated with the specified LIF. The Potential Access CPUs list is controlled through the



access-list attributes of the parent SAC of that PIF. The processors are listed in numerical order.

State

indicates the current state of the LIF.

Trace Filename

indicates the file name that stores the current trace information if TRACE is ON for this LIF.

Trace Status

indicates whether TRACE is active on the specified LIF.

## STATUS MON Command

The STATUS MON command displays the current state of the LANMON processes. The state of a LANMON process can be STARTED, STARTING, or STOPPED.

### Command Syntax

STATUS [ /OUT <i>file-spec</i> / ] MON <i>lanmon-name</i> [ , DETAIL ]
--

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

MON *lanmon-name*

is the name of the LANMON process for which the status is to be returned. The LANMON process name has the form \$ZZLAN.#ZLM *nn*, for example, \$ZZLAN.#ZLM03.

DETAIL

specifies that additional status information is displayed.

## STATUS MON Display

The format of the display for the STATUS MON command without the DETAIL option is:

```
-> STATUS MON $ZZLAN.#ZLM??
```

SLSA Status MON				
Name	State	PID	Priority	Trace Status
\$ZZLAN.#ZLM03	STARTED	( 3,259)	180	OFF
\$ZZLAN.#ZLM02	STARTED	( 2,259)	180	OFF
\$ZZLAN.#ZLM01	STARTED	( 1,266)	180	OFF
\$ZZLAN.#ZLM00	STARTED	( 0,273)	180	OFF

### State

indicates the current state of the LANMON process.

### PID

indicates the processor and process ID of the LANMON process. Is left blank if a LANMON has been aborted.

### Priority

indicates the priority of the LANMON process. Is left blank if a LANMON has been aborted.

### Trace Status

indicates whether tracing has been enabled for the LANMON process.

The format of the display for the STATUS MON command with the DETAIL option is:

```
SLSA Detailed Status MON \SYS.$ZZLAN.#ZLM00
```

Heap Memory Limit.....	133615616
Heap Memory Used.....	69632
PID.....	( 0,273)
Priority.....	200
QIO Pool Current.....	729934
QIO Pool Limit.....	0
State.....	STARTED
Trace Filename.....	
Trace Status.....	OFF

### Heap Memory Limit

indicates the maximum amount of heap space that can be allocated by the LANMON process.

### Heap Memory Used

indicates the current amount of heap space used by the LANMON process.

**PID**

indicates the processor and process ID of the LANMON process. Is left blank if a LANMON has been aborted.

**Priority**

indicates the priority of the LANMON process. Is left blank if a LANMON has been aborted.

**QIO Pool Current**

indicates the current size of the QIO pool used by the LANMON process.

**QIO Pool Limit**

indicates the limit on the amount of QIO pool space that the LANMON process can allocate. 0 indicates no limit.

**State**

indicates the current state of the LANMON process. As of the G06.21 RVU, the summary STOPPING state is now valid for a LANMON process.

**Trace Filename**

indicates the name of the trace destination file if TRACE is ON.

**Trace Status**

indicates if tracing has been enabled for the LANMON process.

## Considerations

In order for the STATUS MON command to accept a wildcard, your system either must have 16 processors or you must precede the STATUS MON command with the SCF command ALLOW ALL ERRORS. See [Verifying the Initial SLSA Subsystem Configuration](#) on page 1-2 for an example of using the ALLOW ALL ERRORS command before using STATUS MON with a wildcard.

## STATUS PIF Command

The STATUS PIF command displays the current summary state of the specified PIF and its current trace status. The summary state of a PIF object can be STARTING, STARTED, or STOPPED.

## Command Syntax

```
STATUS [ /OUT file-spec/ ] PIF pif-name [ , DETAIL ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PIF *pif-name*

specifies the PIF for which the status is to be displayed. The PIF name has the form \$ZZLAN.*adapter-name.sac-unit#.pif-unit-letter*, for example, G1134.0.A.

DETAIL

specifies that additional status information is displayed.

## Considerations

- Interface status information is not returned in the display for the DETAIL option.
- E4SAs, CCSAs, FESAs, GESAs, G4SAs, ATM3SAs, and TRSAs have specific PIF attributes (SWAN does not).

## STATUS PIF Display

The format of the display for the STATUS PIF command for the FESA and CCSA is:

```
-> STATUS PIF $ZZLAN.E*.*

SLSA Status PIF

Name                State    Trace Status
$ZZLAN.CC1.0.A      STARTED
$ZZLAN.F0152.0.A    STARTED    ON
```

State

indicates the current state of the PIF.

Trace Status

indicates whether TRACE is active on the specified PIF.

The format of the display for the STATUS PIF command for the PIF on a TRSA is:

```
-> STATUS PIF $ZZLAN.T0154.*

SLSA Status PIF

Name                State    Trace Status
$ZZLAN.T0154.0.A    STARTED    OFF
```

## STATUS PIF DETAIL for an E4SA

The format of the display for the STATUS PIF command with the DETAIL option for a PIF on an E4SA is:

```
-> STATUS PIF $ZZLAN.E0153.0.A , DETAIL

SLSA Detailed Status PIF \SYS.$ZZLAN.E0153.0.A

CPUs with Data Path..... ( 0 )
Last Error..... (0, 0, 0)
State..... STARTED
Trace Filename.....
Trace Status..... OFF

E4SA Controller Status

Link Pulse State..... UP
```

### CPUs with Data Path

lists processors that have a successful data path for the specified PIF. NONE indicates that no processor has a data path through the PIF. The processors are listed in numerical order.

### Last Error

indicates the severity, origin, and error code of the last error encountered by the PIF. (0,0,0) indicates that no errors have occurred.

### State

indicates the current state of the PIF.

### Trace Filename

indicates the name of the file that stores current trace information if TRACE is ON for the specified PIF.

### Trace Status

indicates whether TRACE is active on the specified PIF.

### Link Pulse State

indicates the current state of the link pulse on the physical Ethernet link. This field is only applicable to the E4SA, FESA, GESA, and G4SA. The value of this field can be either UP (normal condition) or DOWN.

## STATUS PIF DETAIL for a FESA

The format of the display for the STATUS PIF command with the DETAIL option for the PIF on a FESA adapter is:

```
-> STATUS PIF F0152.0.A , DETAIL

CPUs with Data Path..... ( 0, 1 )
Last Error..... (3, LANMON, 2038)
State..... STARTED
Trace Filename.....
Trace Status..... OFF

FESA Controller Status

Duplex..... HALF
Line Speed..... 100 Mbit/sec
Link Pulse State..... Up
```

### CPUs with Data Path

lists processors that have a successful data path for the specified PIF. NONE indicates that no processor has a data path through the PIF. The processors are listed in numerical order.

### Last Error

indicates the severity, origin, and error code of the last error encountered by the PIF. (0,0,0) indicates no errors have occurred.

### State

indicates the current state of the PIF.

### Trace Filename

indicates the name of the file that stores current trace information if TRACE is ON for the specified PIF.

### Trace Status

indicates whether TRACE is active on the specified PIF.

### Duplex

indicates whether the line is full or half duplex. Full duplex means that data can be sent and received at the same time. Half duplex means that data can be sent in only one direction at a time. Invalid means the adapter is not properly connected to the hub.

### Line Speed

indicates the line speed. Possible values are 100 Mbit/sec, 10 Mbit/sec or Invalid. Invalid means the adapter is not properly connected to the hub.

### Link Pulse State

indicates the current state of the link pulse on the physical Ethernet link. This field is only applicable to the E4SA, FESA, GESA, and G4SA. The value of this field can be UP (normal condition) or DOWN.

## STATUS PIF DETAIL for a GESA

The format of the display for the STATUS PIF command with the DETAIL option for the PIF on an GESA adapter is:

```
-> STATUS PIF G0154.0.A , DETAIL

CPUs with Data Path..... ( 0, 1, 2, 3 )
Interface Status . .OFF
Last Error..... (3, LANMON, 2038)
State..... STARTED
Trace Filename.....
Trace Status..... OFF

GESA Adapter Status

Duplex..... FULL
Line Speed..... 1000 Mbit/sec
Link Pulse State..... Up
```

### CPUs with Data Path

lists processors that have a successful data path for the specified PIF. NONE indicates that no processor has a data path through the PIF. The processors are listed in numerical order.

### Last Error

indicates the severity, origin, and error code of the last error encountered by the PIF. (0,0,0) indicates that no errors have occurred.

### State

indicates the current state of the PIF.

### Trace Filename

indicates the name of the file that stores current trace information if TRACE is ON for the specified PIF.

### Trace Status

indicates whether TRACE is active on the specified PIF.

### Duplex

indicates whether the line is full-duplex, half-duplex, or invalid. Full-duplex means that data can be sent and received at the same time. Half-duplex means that data can be sent in only one direction at a time. Invalid means that the adapter is not connected properly to the hub.

### Line Speed

indicates the line speed. Values are 1000 Mbit/sec, 100 Mbit/sec, 10 Mbit/sec or Invalid. Invalid means that the adapter is not connected properly to the hub.

For GESA, LINESPEED defaults to 1000 Mbit/s when the Interface Type is Fiber (SX) or Fiber (LX), even if you specified a LINESPEED of 10 or 100 Mbit/s. (Use the STATUS PIF command to check the actual LineSpeed of the communication line. Use the INFO PIF display to check the user-configured LineSpeed Value.)

### Link Pulse State

indicates the current state of the link pulse on the physical Ethernet link. This field is only applicable to the E4SA, FESA, GESA, and G4SA. The value of this field can be UP (normal condition) or DOWN.

## STATUS PIF DETAIL for a G4SA

The format of the display for the STATUS PIF command with the DETAIL option for the PIF on an G4SA adapter is:

```

SLSA Detailed Status PIF \SYS.$ZZLAN.G11123.0.A

CPUs with Data Path..... ( 0, 1, 2, 3, 4, 5, 6, 7 )
Interface Status..... OFF
Last Error..... (0, 0, 0)
State..... STARTED
Trace Filename.....
Trace Status..... OFF

G4SA Adapter Status

Duplex..... FULL
Line Speed..... 100 Mbit/sec
Link Pulse State..... UP
Line Type ..... Copper
JumboFrame..... OFF
Filter Count..... 27

```

### CPUs with Data Path

lists processors that have a successful data path for the specified PIF. NONE indicates that no processor has a data path through the PIF. The processors are listed in numerical order.

### Last Error

indicates the severity, origin, and error code of the last error encountered by the PIF. (0,0,0) indicates that no errors have occurred.

### State

indicates the current state of the PIF.



### Trace Filename

indicates the name of the file that stores current trace information if TRACE is ON for the specified PIF.

### Trace Status

indicates whether TRACE is active on the specified PIF.

### Duplex

indicates whether the line is full-duplex, half-duplex, or invalid. Full-duplex means that data can be sent and received at the same time. Half-duplex means that data can be sent in only one direction at a time. Invalid means that the adapter is not connected properly to the hub.

### Line Speed

indicates the line speed. Values are 1000 Mbit/sec, 100 Mbit/sec, 10 Mbit/sec or Invalid. Invalid means that the adapter is not connected properly to the hub.

For G4SA, LINESPEED defaults to 1000 Mbit/s when the line type is fiber, even if you specified a LINESPEED of 10 or 100 Megabits/second (Mbit/s). (Use the STATUS PIF command to check the actual LINESPEED of the communication line. Use the INFO PIF display to check the user-configured LINESPEED value.)

### Line Type

indicates the line type, which is either copper or fiber. This field is only displayed for the G4SA adapter type.

### Link Pulse State

indicates the current state of the link pulse on the physical Ethernet link. This field is only applicable to the E4SA, FESA, GESA, and G4SA adapter types. The value of this field can be UP (normal condition) or DOWN.

### JumboFrame

indicates the current state of the link pulse on the physical Ethernet link. This field is only displayed for the G4SA adapter type. The value of this field can be UP (normal condition) or DOWN.

For G4SA, the actual setting of the JumboFrame attribute will be displayed, and jumbo frames cannot be used for line speeds less than 1 Gigabits/second (Gbit/s) even if the JumboFrame attribute has been set to ON. If the INFO PIF, detail command shows JumboFrame ON and the STATUS PIF, detail command shows JumboFrame OFF, the G4SA adapter cannot handle jumbo frames.

### Filter Count

indicates how many filters are registered with the PIF. This field is only displayed for the G4SA adapter type.

## STATUS PIF DETAIL for a TRSA

```
-> STATUS PIF $ZZLAN.T0154.0.A , DETAIL

SLSA Detailed Status PIF \SYS.$ZZLAN.T0154.0.A

CPUs with Data Path..... ( 0 )
Interface Status.....
Last Error..... (0, 0, 0)
State..... STARTED
Trace Filename.....
Trace Status..... OFF

Interface Type .....IP
Duplex.....FULL
Interface Speed.....100,000 Kbit/sec

TRSA Controller Status

Ring State..... OPENED
Ring Open State..... OPEN SUCCESSFUL
Last Ring Status..... NO PROBLEMS
```

### CPUs with Data Path

lists processors that have a successful data path for the PIF associated with the specified LIF. NONE means that no processor has a data path through the PIF. The processors are listed in numerical order.

### Interface Status

indicates the status of the PIF.

### Last Error

indicates the last error encountered by the SAC in the form of (*severity, origin, error code*). (0, 0, 0) indicates no errors have occurred. The possible origins are:

MON	DIH
QIO	LMLIB
NSK	XIO
Filter	SvNet Proc.
SvNet Int.	DSM
NSK Config	LAN Manager
Service Processor	

### State

indicates the current state of the PIF.

### Trace Filename

indicates the name of the trace destination file if trace is on.

### Trace Status

indicates whether tracing has been enabled for the PIF.

### Interface Type

The interface type can be T1, J1, or IP.

### Duplex

The duplex mode can be half or full duplex.

### Interface Speed

The interface speed is in Kbit/sec.

### Ring State

indicates the current state of the token ring. This field is only applicable to the TRSA (Token Ring ServerNet adapter). The value of this field can be one of the following:

OPENED	CLOSED
OPENING	CLOSING
OPEN FAILURE	RING FAILURE

### Ring Open State

indicates the success, or the reason for failure, of the adapter's most recent attempt to enter the token ring. This field is only applicable to the TRSA (Token Ring ServerNet adapter). The value in this field can be one of the following:

NO OPEN ATTEMPTED	OPEN SUCCESSFUL	BAD PARAMETER
LOBE FAILURE	SIGNAL LOSS	INSERTION TIMEOUT
RINGFAILED	BEACONING	DUPLICATE MAC ADDRESS
REQUEST FAILED	REMOVE RECEIVED	

### Last Ring Status

is the current interface status of the token ring. This field is only valid if the Ring State is OPENED. This field is only applicable to the TRSA (Token Ring ServerNet adapter). The value in this field can be one of the following:

NO PROBLEMS	RING RECOVERY	SINGLE STATION
REMOVE RECEIVED	AUTO REMOVAL ERROR	LOBE WIRE FAULT
TRANSMIT BEACON	SOFT ERROR	HARD ERROR
SIGNAL LOSS		

## STATUS PIF DETAIL for an ATM3SA

```
status pif $zzlan.a0253.0.a, detail

SLSA Status DETAIL for PIF \SYS.$ZZLAN.A0253.0.A

CPUs with Data Path.....(0,1)
Last Error.....(0,0,0)
State.....STARTED
Trace Filename.....
Trace Status.....OFF

ATM2 Controller Status

Time Last STARTED.....30 Jul 2001, 14:26:05.791
Time Last STOPPED.....02 Jul 2001, 10:11:56.452
```

### State

indicates the summary state of the PIF object.

### Trace Filename

indicates the name of the file that stores current trace information if TRACE is ON for the specified PIF.

### Trace Status

indicates whether TRACE is active on the specified PIF.

### Time Last STARTED

indicates the time that the PIF last successfully entered the STARTED summary state. This may not be the same time the START PIF command was issued.

### Time Last Stopped

indicates the time the PIF entered the STOPPED summary state. This may not be the same time the STOP PIF or ABORT PIF command was issued.

# STATUS PROCESS Command

The STATUS PROCESS command displays the current state of the LANMAN process. The state of LANMAN can only be STARTED. This is a nonsensitive command.

## Command Syntax

```
STATUS [ /OUT file-spec/] PROCESS $ZZLAN [ , DETAIL ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PROCESS \$ZZLAN

specifies the LANMAN process for which the status is to be displayed. The name of the LANMAN process is \$ZZLAN.

DETAIL

specifies that additional status information is displayed.

## STATUS PROCESS Display

The format of the display for the STATUS PROCESS command without the DETAIL option is:

```
-> STATUS PROCESS $ZZLAN

SLSA Status PROCESS

Name      State      PPID      BPID      Priority  Trace Status
$ZZLAN    STARTED    ( 0, 22)  ( 1, 11)  180      OFF
```

State

indicates the current state of the LANMAN process.

PPID

indicates the primary process and process ID of the LANMAN process.

BPID

indicates the backup process and process ID of the LANMAN process.

Priority

indicates the priority of the LANMAN process.

### Trace Status

indicates whether tracing has been enabled for the LANMAN process.

The format of the display for the STATUS PROCESS command with the DETAIL option is:

```

-> STATUS PROCESS $ZZLAN, DETAIL

SLSA Detailed Status PROCESS \SYS.$ZZLAN
Heap Memory Allocated.... 133615616
Heap Memory Used..... 69632
PID Backup..... ( 1, 11)
PID Primary..... ( 0, 22)
Priority..... 180
QIO Pool Current..... 0
QIO Pool Limit..... 0
State..... STARTED
Trace Filename.....
Trace Status..... OFF

```

### Heap Memory Allocated

indicates the maximum heap memory that can be allocated by the LANMAN process.

### Heap Memory Used

indicates the current heap memory used by the LANMAN process.

### PID Backup

indicates the backup process and process ID of the LANMAN process.

### PID Primary

indicates the primary process and process ID of the LANMAN process.

### Priority

indicates the priority of the LANMAN process.

### QIO Pool Current

indicates the current size of the QIO pool used by the LANMAN process.

### QIO Pool Limit

indicates the limit on the amount of QIO pool space that the LANMAN process can allocate. 0 indicates no limit.

### State

indicates the current state of the LANMAN process.

Trace Filename

indicates the name of the trace destination file if trace is ON.

Trace Status

indicates whether tracing is enabled for the LANMAN process.

## STATUS SAC Command

The STATUS SAC command displays the current state of the specified SAC. The state of a SAC object can be STARTED, STARTING, or STOPPED.

### Command Syntax

```
STATUS [ /OUT file-spec/ ] SAC sac-name [ , DETAIL ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

SAC *sac-name*

is the name of the SAC for which the status is to be returned. The SAC name has the form \$ZZLAN.*adapter-name.sac unit#*, for example, \$ZZLAN.G1123.0.

DETAIL

specifies that additional information is to be displayed.

### Considerations

The last download time field does not return any information in the display for the DETAIL option.

## STATUS SAC Display

The format of the display for the STATUS SAC command is:

```
-> STATUS SAC $ZZLAN.E0153.0
```

SLSA Status SAC			
Name	Owner	State	Trace Status
\$ZZLAN.E0153.0	1	STARTED	ON

Name

is the name of the SAC for which status information is being returned.

Owner

indicates the processor that currently has a data path to the specified SAC.

State

indicates the current state of the SAC specified in the preceding Name field.

Trace Status

indicates whether TRACE is active for the specified SAC.

The format of the display for the STATUS SAC command with the DETAIL option follows. Note that in this display, a LANMON has been ABORTED and the fabric status in the STATUS SAC, DETAIL display indicates that a LANMON is DOWN.

```
status sac s1.1, detail
```

SLSA Detailed Status SAC \SYS.\$ZZLAN.S1.1			
Current Access..... ( 0, 2, 3 )			
Last Download Time..... 20 May 2003, 15:12:53.185			
Last Error..... (0, 0, 0)			
Owner CPU..... 0			
State..... STARTED			
Trace Filename.....			
Trace Status..... OFF			
Fabric Status			
CPU Accesslist	Fabric-X	Fabric-Y	
-----	-----	-----	
0	UP-PRIMARY	UP	
1	UP-PRIMARY	UP	
2	UP-PRIMARY	UP	
3	UP-PRIMARY	UP	

Current Access

lists processors that have successful access to the SAC. NONE indicates that no processor has successful access to the SAC. The processors are listed in numerical order.



### Last Download Time

indicates the last time application microcode was downloaded into the adapter.

### Last Error

indicates the last error encountered by the SAC in the form of (severity, origin, error code). (0, 0, 0) indicates no errors have occurred. The origins can be:

MON	DIH
QIO	LMLIB
NSK	XIO
Filter	SvNet Proc.
SvNet Int.	DSM
NSK Config	LAN Manager
Service Processor	

### Owner CPU

is the processor that currently has ownership of the SAC. NONE means that no processor owns the SAC.

### State

indicates the current state of the SAC.

### Trace Filename

indicates the file name that stores current trace information if TRACE is on for this SAC.

### Trace Status

indicates whether TRACE is active for the specified SAC.

### Fabric Status

indicates the availability of X and Y fabrics for a particular processor in a SAC accesslist. UP-PRIMARY indicates that the fabric is the primary fabric for a particular processor and that it is in the UP state. (For the G4SA adapter type, if the fabric is UP, the fabric status will just be up; unlike other SACs, no primary indication will be shown since the notion of primary path has been eliminated and path choice is made dynamically as long as a path is available.) DOWN indicates that a processor is up but that fabric is not accessible. DISABLED indicates that this fabric is permanently disabled by hardware design. CPU DOWN indicates that the processor has halted or has not been reloaded. MON DOWN indicates that a LANMON has been aborted and the LANMON's associated processor is not available to the X and Y fabrics. The UNKNOWN state is for the extreme case when a fabric state is not in one of the other states, which may indicate a problem with the SAC or ServerNet router.

# STOP Command

STOP is a sensitive command that halts the operation of an object in an orderly manner.

## STOP ADAPTER Command

The STOP ADAPTER command stops the operation of the specified ADAPTER object in an orderly manner and places it in the STOPPED state.

### Command Syntax

```
STOP [ /OUT file-spec/ ] ADAPTER adapter-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*ADAPTER adapter-name*

is the name of the Ethernet adapter to be put in the STOPPED state. The adapter name has the form *\$ZZLAN.adapter-name*, for example, *\$ZZLAN.G1123*.

*SUB [ ONLY | ALL | NONE ]*

directs the command at a set of subordinate objects.

ONLY specifies that only the subordinate objects are affected.

ALL specifies that the named object and the subordinate objects are affected.

NONE specifies that none of the subordinate objects are affected.

### Considerations

- Stopping an adapter does not work when one or more subordinate SAC, LIF, or PIF objects are in the STARTING or STARTED state. You must put them in the STOPPED state before issuing the STOP ADAPTER command, or use the SUB ALL option.
- If a subordinate SAC is in the DIAGNOSE state, you cannot stop the adapter.
- Omitting the SUB option has the same affect as SUB NONE.
- Including the SUB option without specifying ALL, NONE, or ONLY has the same affect as SUB ALL.

## Examples

The following are examples of the STOP ADAPTER command:

```
STOP ADAPTER $ZZLAN.G11123
STOP ADAPTER ($ZZLAN.G11021, $ZZLAN.G11123)
STOP ADAPTER $ZZLAN.G*
STOP ADAPTER $ZZLAN.G11123, SUB
```

## STOP ATMSAP Command

The STOP ATMSAP command requests an ATMSAP object subordinate to a PIF object to halt operation. The ATMSAP object enters the STOPPED summary state if the command is successful. In order for the command to succeed, the LIF object associated with the ATMSAP object must be in the STOPPED summary state or there must not be a LIF object associated with the ATMSAP object. If the ATMSAP object and the LIF object associated with it are both in the STARTED summary state, then STOP ATMSAP will be rejected. An ABORT LIF command must be issued to the LIF object before a STOP ATMSAP command can succeed. The command is rejected if the ATMSAP object is already in the STOPPED summary state.

```
STOP [ /OUT file-spec/ ] ATMSAP atmsap-name
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

ATMSAP *atmsap-name*

is the name of the target ATMSAP object. *atmsap-name* is in the form

```
$ZZLAN.adapter-id.sac-id.pif-id.atmsap-id
```

where:

*adapter-id*

is a 1 to 8-character alpha-numeric string, beginning with an alpha character, that identifies the parent SLSA ADAPTER. Wildcard characters are allowed.

*sac-id*

is a number that identifies the parent SAC object. The only valid *sac-id* for the ATM3SA adapter is the number “0.” Wildcard characters are allowed.

*pif-id*

is an alpha character that identifies the parent PIF object. The only valid *pif-id* for the ATM3SA adapter is the letter “A.” Wildcard characters are allowed.

*atmsap-id*

is a 1 to 8-character alpha-numeric string that identifies the target ATMSAP object. Wildcard characters are allowed.

## Response Display

The STOP ATMSAP command returns only a success or failure indication. Successful completion is indicated when SCF displays the prompt for the next command. A failure is indicated when SCF displays an error message.

## Example

```
STOP ATMSAP $ZZLAN.ATM01.0.A.ATMSAP01
```

## STOP LIF Command

The STOP LIF command stops the operation of the specified LIF and places it in the STOPPED state.

## Command Syntax

```
STOP [ /OUT file-spec/ ] LIF lif-name
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*LIF lif-name*

is the name of the LIF to be put in the STOPPED state. The LIF name has the form *\$ZZLAN.lif-name*, for example, *\$ZZLAN.L11123A*.

## Considerations

- Active clients (registered users) dependent on the LIF can prevent the STOP LIF command from working. Use the ABORT LIF command if you cannot remove the registered users of the specified LIF object.
- You must use ABORT LIF if there are active clients on the LIF.
- For all adapters but MIOF, if the LIF and PIF objects are in STARTED state, it is assumed that there are active clients.
- The STOP LIF command has no effect on the state of the PIF associated with the specified LIF.

- If the LIF and its associated PIF or ATMSAP are both in the STARTED summary state, then STOP LIF will be rejected. An ABORT command must be issued to the PIF or ATMSAP before a STOP LIF command can succeed.

## Examples

The following are examples of the STOP LIF command:

```
STOP LIF $ZZLAN.L11123A
STOP LIF ($ZZLAN.L11123A, $ZZLAN.L11131B)
STOP LIF $ZZLAN.L*
```

## STOP PIF Command

The STOP PIF command halts the operation of the specified PIF in an orderly manner and places it in the STOPPED state.

## Command Syntax

```
STOP [ /OUT file-spec/ ] PIF pif-name
      [ , SUB {ONLY | ALL | NONE } ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PIF *pif-name*

is the name of the PIF to be put in the STOPPED state. The PIF name has the form \$ZZLAN.*pif-name*, for example, \$ZZLAN.11123.0.B.

SUB { ONLY | ALL | NONE }

controls the set of objects and subordinate objects the command targets:

ONLY

specifies that only the subordinate objects are targets of the command.

ALL

specifies that the named object and the subordinate objects are targets of the command. This is the default used if the SUB keyword is used but no option is selected.

NONE

specifies that none of the subordinate objects are targets of the command. This is the default selected if the SUB keyword is not used.

## Considerations

- An active client (registered user) dependent on the PIF (through a LIF associated with the specified PIF) can prevent the STOP PIF command from working. Use the ABORT PIF command if you cannot remove the client dependent on the specified PIF object.
- For E4SAs, FESAs, GESAs, G4SAs, ATM3SAs, and TRSAs, active clients are assumed to exist if the PIF and LIF objects are in the STARTED state.
- Stopping a SAC does not cause subordinate PIFs to stop unless you use the SUB ALL option. You must stop the subordinate PIFs before you can stop the SAC. Also, when the SAC is restarted, the subordinate PIFs are not restarted automatically unless you use the SUB ALL option; otherwise, you must issue START PIF commands to restart the PIFs.
- The STOP PIF command causes the access state of its associated LIF to change to DOWN.
- For adapters other than ATM3SA, if the PIF and the LIF associated with it are both in the STARTED state, the STOP PIF will be rejected.

## Examples

The following are examples of the STOP PIF command:

```
STOP PIF $ZZLAN.G11123.0.B
```

```
STOP PIF ($ZZLAN.G11123.0.A, $ZZLAN.G11123.0.B)
```

```
STOP PIF $ZZLAN.G11123.*
```

## STOP SAC Command

The STOP SAC command stops the operation of the specified SAC and places it in the STOPPED state if there are no active data paths to the specified SAC. Use the ABORT SAC command to force a stop if there are active data paths.

### Command Syntax

```
STOP [ /OUT file-spec/ ] SAC sac-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

OUT

causes any SCF output generated for this command to be directed to the specified file.

SAC *sac-name*

is the name of the ServerNet Addressable controller (SAC) to be stopped. The SAC name has the form `$ZZLAN.adapter-name.sac-unit#`, for example, `$ZZLAN.G11123.0`.

SUB [ ONLY | ALL | NONE ]

directs the command at a set of subordinate objects.

ONLY specifies that only the subordinate objects are affected.

ALL specifies that the named object and the subordinate objects are affected.

NONE specifies that none of the subordinate objects are affected.

If you omit ONLY, ALL, or NONE, ALL is assumed.

### Considerations

- Objects subordinate to the SAC must be stopped before the STOP command works. Use the ABORT SAC command if you cannot stop the objects subordinate to the specified SAC.
- Stopping a SAC does not cause the subordinate PIFs to be stopped unless you use the SUB option. When the SAC is restarted, the subordinate PIFs are not restarted automatically unless you use the SUB ALL option; otherwise, you must issue START PIF commands to restart the PIF objects.
- Omitting the SUB option has the same affect as SUB NONE.
- Including the SUB option without specifying ALL, NONE, or ONLY has the same affect as SUB ALL.

## Examples

The following is an example of the STOP SAC command:

```
STOP SAC $ZZLAN.G11123.0
```

# SWITCH Command

SWITCH is a sensitive command that switches the operation of a LAN Manager (LANMAN) process from one LANMAN to another.

## SWITCH PROCESS Command

The SWITCH PROCESS command switches the operation of the specified LAN Manager (LANMAN) and places the LANMAN in the STOPPED state.

## Command Syntax

```
SWITCH [ /OUT file-spec/ ] PROCESS lanman
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*PROCESS lanman*

is the name of the LAN Manager (LANMAN) process to be switched. The LANMAN name has the form \$ZZLAN, for example, \$ZZLAN.#ZLM01.

## Considerations

A SLSA client may be prevented from switching if the LANMON in which the client's backup is running has been aborted.

## Examples

The following example switches the \$ZZLAN LANMAN process:

```
SWITCH PROCESS $ZZLAN
```



# TRACE Command

TRACE is a sensitive command that initiates trace-data collection on the specified object. You can use traces as a diagnostic tool.

An SCF trace produces a trace file that can be displayed by using the commands available in the PTrace program. The Subsystem Control Point (SCP) creates the trace file. Refer to the *PTrace Reference Manual* for detailed information about PTrace.

## TRACE MON Command

The TRACE MON command executes a trace on the LANMON process.

### Command Syntax

```
TRACE [ /OUT file-spec/ ] MON lanmon-name

    { , STOP }
    [ , BULKIO | NOBULKIO ]
    [ , COUNT count ]
    [ , LOCKSIZE locksize ]
    [ , NOCOLL ]
    [ , PAGES pages ]
    [ , RECSIZE size ]
    [ , SELECT select-spec ]
    [ , TO file-spec ]
    [ , WRAP ] }
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

MON *lanmon-name*

specifies the LANMON process on which a TRACE is to be performed. LANMON names have the form \$ZZLAN.#ZLMnn where *nn* indicates the particular processor in which the LANMON is running.

STOP

discontinues the trace currently in progress.

BULKIO | NOBULKIO

designates whether TRACE should use bulk I/O for tracing. BULKIO specifies that the TRACE collector use bulk I/O to write data to the disk file, reducing the number of missing frame errors reported by PTrace.

A limitation of bulk I/O is that only one user can access the file at a time. If shared access of the trace file is needed while the trace is active, specify NOBULKIO.

Bulk I/O tracing is faster and should eliminate most data loss. The default is BULKIO. BULKIO cannot be used with NOCOLL.

COUNT *count*

specifies the number of trace records to be captured. *count* is an integer in the range -1 through 32767. If *count* is omitted or if *count* equals -1, records are accumulated until the trace is stopped.

LOCKSIZE *locksize*

designates how much memory space, in units of pages, is locked down at one time. LOCKSIZE can be specified only when a trace is initiated. *locksize* is an integer in the range 4 through 1024. *locksize* must be less than or equal to *pages*. If PAGES is not specified, *locksize* must be less than or equal to 64. If LOCKSIZE is omitted, *locksize* is the lesser of *pages* and 64.

NOCOLL

specifies that trace data from the extended segment is written to the disk file specified in TO *file-spec* only when one of the following occurs:

- The trace is stopped.
- The number of trace records written to the extended data segment equals the count specified in the TRACE command that started the trace. When the trace facility detects this condition, the trace records from the extended data segment are written to the disk file and the trace is stopped automatically. You don't need to issue a separate TRACE command to stop the trace. If NOCOLL is not specified, a trace collector process reads the trace records from the extended data segment and writes them to the disk file as they become available.

The NOCOLL option cannot be used with the BULKIO option.

PAGES *pages*

designates how much space, in units of pages, is allocated in the extended data segment used for tracing. PAGES can be specified only when a trace is being initiated. *pages* is an integer in the range 4 through 1024 or is equal to 0. If PAGES is omitted or *pages* equals 0, the default value of 64 pages is assumed.

RECSIZE *size*

specifies the length of the data in the trace-data records. You can specify *size* as 0 or as an integer in the range 16 through 4050. The length of the trace header, which is eight bytes, is not included in *size*. If RECSIZE is omitted or if *size* equals 0, a default value of 120 bytes is assumed.

SELECT *select-spec*

*select-spec* is one of the following specifications:

```
{ keyword }
{ ( keyword [ , keyword ] ) }
{ number }
{ ( number [ , number ] ) }
```

*keyword*

is subsystem-specific. See [Table 4-4](#) for the list of keywords that can be used for the MONITOR objects.

*number*

is the numeric value that a keyword represents. See [Table 4-4](#) for the list of numbers that can be used for the MONITOR objects.

TO *file-spec*

specifies the file to which trace information is to be written. The file might have been previously created as an unstructured file with file code 830. An old file is purged of data before the trace is initiated. If the file does not exist, a file is created with an extent size based upon the number of pages specified in the PAGES option.

WRAP

specifies that when the trace disk file end-of-file (EOF) mark is reached, trace data wraps around to the beginning of the file and overwrites any data there.

**Table 4-4. *select-spec* for a MONITOR object**

Keyword	Number	Meaning
ALL	-1	Trace all items
SMACH	0	Trace state machines
EVT	1	Trace events
QUEUE	2	Trace internal queues
TIMER	3	Trace timer activity
MSG	4	Trace messages
INTMSG	5	Trace internally generated messages
DIH	6	Trace calls to the driver interrupt handler (DIH) routines
MEMORY	7	Trace management of resources for internal memory for LANMON

## Considerations

- If TO *file-spec* is specified, a new trace is initiated unless the *file-spec* is invalid, the file cannot be opened, or trace is already active for the monitor process.
- If TO *file-spec* and STOP are both omitted, the TRACE command modifies the trace currently in progress, if any.
- If TO *file-spec* is omitted and STOP is specified, the TRACE command stops the trace currently in progress, if any.

## Examples

The following example starts a trace for all traceable items in \$ZZLAN.#ZLM01:

```
TRACE MON $ZZLAN.#ZLM01, SELECT ALL, TO $M2.SUBV.TRFIL1
```

The following example traces state-machine and event information in \$ZZLAN.#ZLM01:

```
TRACE MON $ZZLAN.#ZLM01, SELECT (SMACH, EVT), &  
TO $M2.SUBV.TRFIL1
```

The following example stops the trace in \$ZZLAN.#ZLM01:

```
TRACE MON $ZZLAN.#ZLM01, STOP
```

## TRACE PIF Command

The TRACE PIF command executes a trace on a specified PIF. The TRACE operation can increase the LAN DIH use of the processor; therefore, use the TRACE PIF command with caution.

## Command Syntax

```
TRACE [ /OUT file-spec/ ] PIF pif-name  
  
    {  
    [ , STOP ]  
    { [ , BULKIO | NOBULKIO ]  
    [ , COUNT count ]  
    [ , CPU cpu-number ]  
    [ , LOCKSIZE locksize ]  
    [ , NOCOLL ]  
    [ , PAGES pages ]  
    [ , RECSIZE size ]  
    [ , SELECT select-spec ]  
    [ , TO file-spec ]  
    [ , WRAP ]  
    }
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PIF *pif-name*

is the name of the PIF.

STOP

discontinues the trace currently in progress.

BULKIO | NOBULKIO

designates whether TRACE should use bulk I/O for tracing. BULKIO specifies that the TRACE collector use bulk I/O to write data to the disk file, reducing the number of missing frame errors reported by PTrace.

A limitation of BULKIO is that only one user can access the file at a time. If shared access of the trace file is needed while the trace is active, specify NOBULKIO. Bulk I/O tracing is faster and should eliminate most data loss. The default is BULKIO. BULKIO cannot be used with NOCOLL.

COUNT *count*

specifies the number of trace records to be captured. *count* is an integer in the range -1 through 32767. If COUNT is omitted or if *count* equals -1, records are accumulated until the trace is stopped.

CPU *cpu-number*

specifies the CPU number from which the trace records are to be captured. *cpu-number* is an integer in the range 0 through 15. You can only specify the CPU that is on the parent SAC's ACCESSLIST, and the associated CPU must be loaded. If the CPU attribute is not specified, the owner CPU of the SAC is traced.

If the specified CPU is down at the time the TRACE command is issued, the system will return an error and an empty trace file will be generated.

LOCKSIZE *locksize*

designates how much memory space, in units of pages, is locked down at one time. LOCKSIZE can be specified only when a trace is initiated. *locksize* is an integer in the range 4 through 1024. *locksize* must be less than or equal to *pages*. If PAGES is not specified, *locksize* must be less than or equal to 64. If LOCKSIZE is omitted, *locksize* is the lesser of *pages* or 64.

NOCOLL

specifies that trace data from the extended segment is written to the disk file specified in TO *file-spec* only when one of the following occurs:

- The trace is stopped.
- The number of trace records written to the extended data segment is equal to the count specified in the TRACE command that started the trace. When the trace facility detects this condition, the trace records from the extended data

segment are written to the disk file and the trace is stopped automatically. You do not need to issue a separate TRACE command to stop the trace. If NOCOLL is not specified, a trace collector process reads the trace records from the extended data segment and writes them to the disk file as they become available.

The NOCOLL option cannot be used with the BULKIO option.

#### PAGES *pages*

designates how much space, in units of pages, is allocated in the extended data segment used for tracing. PAGES can be specified only when a trace is being initiated. *pages* is an integer in the range 4 through 1024 or is equal to 0. If PAGES is omitted or *pages* equals 0, the default value of 64 pages is assumed.

#### RECSIZE *size*

specifies the length of the data in the trace-data records. *size* is an integer in the range 16 through 4050, and 0. The length of the trace header (8 bytes) is not included in *size*. If RECSIZE is omitted or if *size* equals 0, the default value of 120 bytes is assumed.

#### SELECT *select-spec*

*select-spec* is one of the following specifications:

```
{ keyword }
{ ( keyword [ , keyword ] ) }
{ number }
{ ( number [ , number ] ) }
```

#### keyword

is subsystem-specific. See [Table 4-5](#) for the list of keywords that can be used with PIF objects.

#### number

is the numeric value that a keyword represents. See [Table 4-5](#) for the list of numbers that can be used for the PIF objects.

#### TO *file-spec*

specifies the file to which trace information is to be written. The file might have been previously created as an unstructured file using file code 830. An old file is purged of data before the trace is initiated. If the file does not exist, a file is created with an extent size based upon the number of PAGES specified.

#### WRAP

specifies that when the trace disk file end-of-file (EOF) mark is reached, trace data wraps around to the beginning of the file and overwrites any data there.

**Table 4-5. *select-spec* for a PIF object**

Keyword	Number	Meaning
ALL	-1	Trace all items
CALLIN	8	Trace call in from external components for PIF object in DIH
CALLOUT	9	Trace call out to external components for PIF object in DIH
CALLLOCAL	10	Trace local call for PIF object in DIH
PING	11	Trace PIF object ping related action in DIH
SMACH	12	Trace PIF state machine operations for PIF object
SNET	13	Trace PIF object ServerNet events
SNETINT	14	Trace PIF object ServerNet interrupt events
QSERV	15	Trace PIF object queue service
TXDATA	16	Trace PIF object outbound data
RXDATA	17	Trace PIF object inbound data
LMOMSG	18	Trace messages between LANMON and DIH for PIF objects
ERROR	19	Trace DIH errors for PIF object

## Considerations

- You cannot execute a trace on a PIF until its parent SAC is in the STARTED state. The trace on the PIF terminates automatically when:
  - The parent SAC is no longer in the STARTED state.
  - The data path to the SAC switched to a different processor; that is, an ownership change occurs.
  - The trace file becomes full, and you did not specify the WRAP option when the trace was initiated.
- You can have a maximum of 64 open traces for SAC, PIF, and LIF objects in a single processor.
- If TO *file-spec* is specified, a new trace is initiated unless the *file-spec* is invalid, the file cannot be opened, or a trace is already active for the PIF.
- If TO *file-spec* and STOP are both omitted, the TRACE command modifies the trace currently in progress, if any.
- If TO *file-spec* is omitted and STOP is specified, the TRACE command stops the trace currently in progress, if any.

## Example

This example traces DIH state-machine operations and messages between a LANMON and DIH for the PIF named G11123.0.B. The maximum length of the traced-data record is 2048 bytes. The information is saved in the file named TRFIL1.

```
TRACE PIF $ZZLAN.$ZZLAN.G11123.0.B, RECSIZE 2048, &
SELECT (SMACH, LMOMSG), TO $M2.SUBV.TRFIL1
```

## TRACE PROCESS Command

The TRACE PROCESS command initiates a trace on the specified LANMAN process.

## Command Syntax

```
TRACE [ /OUT file-spec/ ] [PROCESS process-name ]
    { , STOP }
    { , [ , BULKIO | NOBULKIO ]
      [ , COUNT count ]
      [ , LOCKSIZE locksize ]
      [ , NOCOLL ]
      [ , PAGES pages ]
      [ , RECSIZE size ]
      [ , SELECT select-spec ]
      [ , TO file-spec ]
      [ , WRAP ] }
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

PROCESS *process-name*

specifies the process for which a trace is to be initiated. For the LANMAN process, *process-name* is \$ZZLAN.

STOP

discontinues the trace currently in progress.

BULKIO | NOBULKIO

designates whether TRACE should use bulk I/O for tracing. BULKIO specifies that the TRACE collector use bulk I/O to write data to the disk file, reducing the number of missing frame errors reported by PTrace.

A limitation of BULKIO is that only one user can access the file at a time. If shared access of the trace file is needed while the trace is active, specify NOBULKIO. Bulk I/O tracing is faster and should eliminate most data loss. The default is BULKIO. BULKIO cannot be used with NOCOLL.



**COUNT** *count*

specifies the number of trace records to be captured. *count* is an integer in the range -1 through 32767. If COUNT is omitted or if *count* equals -1, records are accumulated until the trace is stopped.

**LOCKSIZE** *locksize*

designates how much memory space, in units of pages, is locked down at one time. LOCKSIZE can be specified only when a trace is initiated. *locksize* is an integer in the range 4 through 1024. *locksize* must be less than or equal to *pages*. If PAGES is not specified, *locksize* must be less than or equal to 64. If LOCKSIZE is omitted, *locksize* is the lesser of *pages* or 64.

**NOCOLL**

specifies that trace data from the extended segment is written to the disk file specified in TO *file-spec* only when one of the following occurs:

- The trace is stopped.
- The number of trace records written to the extended data segment is equal to the count specified in the TRACE command that started the trace. When the trace facility detects this condition, the trace records from the extended data segment are written to the disk file and the trace is stopped automatically. You don't need to issue a separate TRACE command to stop the trace. If NOCOLL is not specified, a trace collector process reads the trace records from the extended data segment and writes them to the disk file as they become available.

The NOCOLL option cannot be used with the BULKIO option.

**PAGES** *pages*

designates how much space, in units of pages, is allocated in the extended data segment used for tracing. PAGES can be specified only when a trace is being initiated. *pages* is an integer in the range 4 through 1024 or is equal to 0. If PAGES is omitted or if *pages* equals 0, the default value of 64 pages is assumed.

**RECSIZE** *size*

specifies the length of the data in the trace-data records. *size* is an integer in the range 16 through 4050, and 0. The length of the trace header, which is 8 bytes, is not included in *size*. If RECSIZE is omitted or if *size* equals 0, the default value of 120 bytes is assumed.

SELECT *select-spec*

*select-spec* is one of the following specifications:

```
{ keyword }
{ ( keyword [ , keyword ] ) }
{ number }
{ ( number [ , number ] ) }
```

*keyword*

is subsystem-specific. ALL is the only keyword that you can use with the TRACE PROCESS command.

*number*

is the numeric value that a keyword represents. The numeric value of ALL is -1.

TO *file-spec*

specifies the file to which trace information is to be written. The file might have been previously created as an unstructured file with file code 830. An old file is purged of data before the trace is initiated. If the file does not exist, a file is created with an extent size based upon the number of pages specified in the PAGES option.

WRAP

specifies that when the trace disk file end-of-file (EOF) mark is reached, trace data wraps around to the beginning of the file and overwrites any data there.

## Considerations

- If TO *file-spec* is specified, a new trace is initiated unless the *file-spec* is invalid, the file cannot be opened, or trace is already active for the I/O process.
- If TO *file-spec* and STOP are both omitted, the TRACE command modifies the trace currently in progress, if any.
- If TO *file-spec* is omitted and STOP is specified, the TRACE command stops the trace currently in progress, if any.

## Example

The following example starts a trace for all traceable items in \$ZZLAN:

```
TRACE PROCESS $ZZLAN, TO $M2.SUBV.TRFIL1
```

## TRACE SAC Command

The TRACE SAC command traces the SAC object in a processor. The TRACE operation can significantly increase the LAN DIH use of the processor; therefore, use the TRACE SAC command with caution. This is a sensitive command.

### Command Syntax

```
TRACE [ /OUT file-spec/ ] SAC sac-name

    {
      , STOP
    [ , BULKIO | NOBULKIO ]
    [ , COUNT count ]
    [ , CPU cpu-number ]
    [ , LOCKSIZE locksize ]
    [ , NOCOLL ]
    [ , PAGES pages ]
    [ , RECSIZE size ]
    [ , SELECT select-spec ]
    [ , TO file-spec ]
    [ , WRAP ]
    }
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

SAC *sac-name*

is the name of the SAC.

STOP

discontinues the trace currently in progress.

BULKIO | NOBULKIO

designates whether TRACE should use bulk I/O for tracing. BULKIO specifies that the TRACE collector use bulk I/O to write data to the disk file, reducing the number of missing frame errors reported by PTrace.

A limitation of BULKIO is that only one user can access the file at a time. If shared access of the trace file is needed while the trace is active, specify NOBULKIO. Bulk I/O tracing is faster and should eliminate most data loss. The default is BULKIO. BULKIO cannot be used with NOCOLL.

COUNT *count*

specifies the number of trace records to be captured. *count* is an integer in the range -1 through 32767. If COUNT is omitted or if *count* equals -1, records are accumulated until the trace is stopped.

**CPU** *cpu-number*

specifies the CPU number from which the trace records are to be captured.

*cpu-number* is an integer in the range 0 through 15. You can only specify the CPU that is on the parent SAC's ACCESSLIST, and the associated CPU must be loaded. If the CPU attribute is not specified, the owner CPU of the SAC is traced.

If the specified CPU is down at the time the TRACE command is issued, the system will return an error and an empty trace file will be generated.

**LOCKSIZE** *locksize*

designates how much memory space, in units of pages, is locked down at one time. LOCKSIZE can be specified only when a trace is initiated. *locksize* is an integer in the range 4 through 1024. *locksize* must be less than or equal to *pages*. If PAGES is not specified, *locksize* must be less than or equal to 64. If LOCKSIZE is omitted, *locksize* is the lesser of *pages* or 64.

**NOCOLL**

specifies that trace data from the extended segment is written to the disk file specified in TO *file-spec* only when one of the following occurs:

- The trace is stopped.
- The number of trace records written to the extended data segment is equal to the count specified in the TRACE command that started the trace. When the trace facility detects this condition, the trace records from the extended data segment are written to the disk file and the trace is stopped automatically. You don't need to issue a separate TRACE command to stop the trace. If NOCOLL is not specified, a trace collector process reads the trace records from the extended data segment and writes them to the disk file as they become available.

NOCOLL option cannot be used with the BULKIO option.

**PAGES** *pages*

designates how much space, in units of pages, is allocated in the extended data segment used for tracing. PAGES can be specified only when a trace is being initiated. *pages* is an integer in the range 4 through 1024 or is equal to 0. If PAGES is omitted or if *pages* equals 0, the default value of 64 pages is assumed.

**RECSIZE** *size*

specifies the length of the data in the trace-data records. *size* is an integer in the range 16 through 4050, and 0. The length of the trace header, which is 8 bytes, is not included in *size*. If RECSIZE is omitted or if *size* equals 0, the default value of 120 bytes is assumed.

SELECT *select-spec*

*select-spec* is one of the following specifications:

```
{ keyword }
{ ( keyword [ , keyword ] ) }
{ number }
{ ( number [ , number ] ) }
```

*keyword*

is subsystem-specific. ALL is the only keyword that you can use with the TRACE PROCESS command.

*number*

is the numeric value that a keyword represents. The numeric value of ALL is -1.

TO *file-spec*

specifies the file to which trace information is to be written. The file might have been previously created as an unstructured file with file code 830. An old file is purged of data before the trace is initiated. If the file does not exist, a file is created with an extent size based upon the number of pages specified in the PAGES option.

WRAP

specifies that when the trace disk file end-of-file (EOF) mark is reached, trace data wraps around to the beginning of the file and overwrites any data there.

**Table 4-6. *select-spec* for a SAC object** (page 1 of 2)

Keyword	Number	Meaning
ALL	-1	Trace all items
CALLIN	8	Trace call in from external components for SAC object in DIH
CALLOUT	9	Trace call out to external components for SAC object in DIH
CALLLOCAL	10	Trace local call for SAC object in DIH
PING	11	Trace SAC object ping related action in DIH
SMACH	12	Trace DIH state machine operations for SAC object
SNET	13	Trace SAC object ServerNet events
SNETINT	14	Trace SAC object ServerNet interrupt events
QSERV	15	Trace SAC object queue service
TXDATA	16	Trace SAC object outbound data

**Table 4-6. *select-spec* for a SAC object** (page 2 of 2)

Keyword	Number	Meaning
RXDATA	17	Trace SAC object inbound data
LMOMSG	18	Trace messages between LANMON and DIH for SAC objects
ERROR	19	Trace DIH errors for SAC object

## Considerations

- You can initiate or terminate a SAC while it is in any state. The trace on a SAC is terminated automatically when:
  - It is deleted along with its parent adapter.
  - Its ownership switches to a different processor.
  - The trace file becomes full and no WRAP option has been specified when the trace was initiated.
- You can have a maximum of 64 open traces for SAC, PIF, and LIF objects in a single processor.
- If TO *file-spec* is specified, a new trace is initiated unless the *file-spec* is invalid, the file cannot be opened, or trace is already active for the SAC.
- If TO *file-spec* and STOP are both omitted, the TRACE command modifies the trace currently in progress, if any.
- If TO *file-spec* is omitted and STOP is specified, the TRACE command stops the trace currently in progress, if any.

## Example

This example traces DIH state machine operations and messages between a LANMON and DIH for the SAC named G11123.0. The maximum length of the traced-data record is 2048 bytes. The information is saved in the file named TRFIL1.

```
TRACE SAC $ZZLAN.G11123.0, RECSIZE 2048, &
SELECT (SMACH, LMOMSG), TO $M2.SUBV.TRFIL1
```

# VERSION Command

The VERSION command is a nonsensitive command that displays version information about the SLSA subsystem and the level of support for security and tracing.

## VERSION MON Command

The VERSION MON command displays the version information of the LANMON monitor processes. Use the DETAIL option to display version information about the SCF Kernel and the SCF SLSA product module.

This is a nonsensitive command.

## Command Syntax

```
VERSION [ /OUT file-spec/ ] MON lanmon-name [ , DETAIL ]
```

OUT *file-spec*

causes any SCF output generated for this command to be directed to the specified file.

MON *lanmon-name*

specifies the LANMON for which version information is to be displayed. LANMON names have the form \$ZZLAN.#ZLMnn where *nn* indicates the particular processor in which the LANMON is running.

DETAIL

specifies that additional information is to be displayed.

## VERSION MON Display

The format of the display for the VERSION MON command without the DETAIL option is:

```
-> VERSION MON $ZZLAN.*

VERSION MON \SYS.$ZZLAN.#ZLM00: LANMonitor -
T8174G02_01MAY97_28FEB97_MON_V2
VERSION MON \SYS.$ZZLAN.#ZLM01: LANMonitor -
T8174G02_01MAY97_28FEB97_MON_V2
VERSION MON \SYS.$ZZLAN.#ZLM02: LANMonitor -
T8174G02_01MAY97_28FEB97_MON_V2
VERSION MON \SYS.$ZZLAN.#ZLM03: LANMonitor -
T8174G02_01MAY97_28FEB97_MON_V2
```

The format of the display for the VERSION MON command with the DETAIL option is:

```
-> VERSION MON $ZZLAN.#ZLM01 , DETAIL

Detailed VERSION MON \SYS.$ZZLAN.#ZLM01
SYSTEM \SYS
  LANMonitor - T8174G02_01MAY97_28FEB97_MON_V2
  GUARDIAN - T9050 - (P40)
  SCF KERNEL - T9082F40 - (29FEB96) (01JAN96)
  SLSA PM - T7894G02 - (01MAY97) (01MAR97) - (SLSA) (V1)
```

## VERSION null Command

The VERSION null command displays the version information of the LANMON process. Use the DETAIL option to display version information about the SCF Kernel and the SCF SLSA product module.

### Command Syntax

```
VERSION [/OUT file-spec/ ] [ object-name ] [ , DETAIL ]
```

*OUT file-spec*

causes any SCF output generated for this command to be directed to the specified file.

*object-name*

is the name of the LANMAN process, \$ZZLAN. Omitting the object-name invokes the null object.

DETAIL

specifies that additional information is to be displayed.

### Considerations

You must enter the name of the LAN manager process (\$ZZLAN) with the VERSION null command.

### VERSION null Display

The format of the display for the VERSION null command is:

```
-> VERSION $ZZLAN

VERSION : LANManager - T8173G06_19JUL99_02JUL99AAJ
```



The format of the display for the VERSION null command with the DETAIL option is:

```
-> VERSION $ZZLAN , DETAIL

Detailed VERSION
SYSTEM \SYS
  LANManager - T8173G06_19JUL99_02JUL99AAJ
  GUARDIAN - T9050 - (Q06)
  SCF KERNEL - T9082G02 - (24SEP99) (26JUL99)
  SLSA PM - T7894G06 - (19JUL99) (11JUN99) - (SCF) (V1)
```

## VERSION PROCESS Command

The VERSION PROCESS command displays the version information for the LANMAN process.

### Command Syntax

```
VERSION [ /OUT file-spec/ ] PROCESS process-name [ , DETAIL
]
```

OUT *file-spec*

causes any SCF out generated for this command to be directed to the specified file.

PROCESS *process-name*

specifies the process for version information is to be displayed. For the LANMAN process, *process-name* is \$ZZLAN.

DETAIL

displays version information about the SCF Kernel and the SCF SLSA product module.

### VERSION PROCESS Display

The format of the display for the VERSION PROCESS command without the DETAIL option is:

```
-> VERSION PROCESS $ZZLAN

VERSION PROCESS \SYS.$ZZLAN: LANManager - T8173G06_19JUL99_02JUL99AA
```

The format of the display for the VERSION PROCESS command with the DETAIL option is:

```
-> VERSION PROCESS $ZZLAN , DETAIL

Detailed VERSION PROCESS \SYS.$ZZLAN
SYSTEM \SYS
  LANManager - T8173G06_19JUL99_02JUL99AAJ
  GUARDIAN - T9050 - (Q06)
  SCF KERNEL - T9082G02 - (24SEP99) (26JUL99)
  SLSA PM - T7894G06 - (19JUL99) (11JUN99) - (SCF) (V1)
```

# Managing the SLSA Subsystem

This section describes management and failure recovery for the SLSA subsystem. The following management tasks are described in this section:

<a href="#">Adding an Adapter to the System</a>	<a href="#">5-1</a>
<a href="#">Stopping and Starting an Adapter</a>	<a href="#">5-4</a>
<a href="#">Renaming an Adapter</a>	<a href="#">5-7</a>
<a href="#">Altering the Access List</a>	<a href="#">5-8</a>
<a href="#">Altering the LANMAN Process</a>	<a href="#">5-9</a>
<a href="#">Aborting the LANMAN Process</a>	<a href="#">5-9</a>
<a href="#">Replacing a LANMON Without a Cold Load</a>	<a href="#">5-10</a>

## Adding an Adapter to the System

---

**Note.** WAN Wizard Pro can be used as an alternative to Subsystem Control Facility (SCF) to configure LAN adapters. For more information, see [WAN Wizard Pro](#) on page 1-8.

---

- 
- △ **Caution.** NonStop TCP/IPv6 and Parallel Library TCP/IP are incompatible and cannot run on the same system. Parallel Library TCP/IP is not supported on Integrity NonStop NS-series servers.
- 

When you add an adapter to a NonStop system, you specify:

- The name you want to use to identify the adapter; for example, E0153 for an Ethernet 4-port ServerNet adapter (E4SA) in slot 53 of the first system enclosure or G11123 for a Gigabit Ethernet 4-port ServerNet adapter (G4SA) in group 111, module 2, and slot 3 of an I/O adapter module (IOAM) enclosure.
- The type of LAN adapter you are adding to the system: ATM3SA, CCSA, E4SA, FESA, G4SA, GESA, MFIOB, or TRSA. G4SAs can be added to an IOAM enclosure or can represent the Ethernet ports in a VIO enclosure. (VIO enclosures are supported in H06.08 and subsequent H-series RVUs).
- The location of the adapter within the enclosure (group, module, and slot).
- The processors that have access to the SACs on the adapter.

To add an adapter to the SLSA subsystem:

1. Make sure the adapter you are adding is supported for the NonStop system type as described in [Supported Adapters by NonStop System Type](#) on page 2-13.
2. Fill out a configuration form for the adapter as described in [Section 3, SLSA Subsystem Installation and Configuration](#).

3. Install the adapter in the system (see [Adapter Manuals](#) on page 1-7).
4. Use the SCF ADD ADAPTER command to add the adapter to the SLSA subsystem. For example:

```
ADD ADAPTER $ZZLAN.G11021, TYPE G4SA, LOCATION (110,2,1), &
ACCESSLIST (0, 1, 2, 3)
```

In this example, SCF add a G4SA to slot 1 of IOAM group 110, module 2, and gives processor 0 primary access to the ServerNet addressable controllers (SACs). Processors 1 through 3 are assigned secondary access. (For a detailed description, see the [ADD ADAPTER Command](#) on page 4-15.)

The system assigns SAC and physical interface (PIF) names derived from the adapter name you used in the ADD command. The SAC(s) on the adapter you add have the same processors as the adapter assigned to them. You can alter the access list for the SAC on an adapter by using the SCF ALTER SAC command. (See [Altering the Access List](#), later in this section and the [ALTER SAC Command](#) on page 4-36 for a detailed description of this command.)

Once the subsystem has added the adapter, the adapter and its subordinate SACs and PIFs are still in the STOPPED state.

---

**Note.** For Token Ring ServerNet adapters (TRSAs), you might need to alter the PIF before starting the adapter and its subordinate objects. Refer to the *Token-Ring Adapter Installation and Support Guide* for information on configuring a TRSA.

---

5. Use the SCF START ADAPTER command with the SUB ALL option to start the adapter and its subordinate SAC(s) and PIFs.

```
START ADAPTER $ZZLAN.G11021, SUB ALL
```

The system starts the SAC(s) and subordinate PIFs on the specified adapter.

6. Use the SCF NAMES command to display the names assigned to the SAC and PIFs of the adapter you added in step 3.

```
NAMES PIF $ZZLAN.G11021*

SLSA Names PIF \SYS.$ZZLAN.G11021

PIF
$ZZLAN.G11021.0.A  $ZZLAN.G11021.0.A
$ZZLAN.G11021.0.B  $ZZLAN.G11021.0.B
$ZZLAN.G11021.0.C  $ZZLAN.G11021.0.C
$ZZLAN.G11021.0.D  $ZZLAN.G11021.0.D
```

7. Assign logical interfaces (LIFs) to the PIFs on the adapter you added in step 3 by using the SCF ADD LIF command. Use the names of the PIFs returned from the system in step 5.

```
ADD LIF $ZZLAN.L11021A, PIF G11021.0.A
ADD LIF $ZZLAN.L11021B, PIF G11021.0.B
ADD LIF $ZZLAN.L11021C, PIF G11021.0.C
ADD LIF $ZZLAN.L11021D, PIF G11021.0.D
```

8. Start the LIFs by using the START LIF command.

```
START LIF $ZZLAN.L11021A
START LIF $ZZLAN.L11021B
START LIF $ZZLAN.L11021C
START LIF $ZZLAN.L11021D
```

---

**Note.** To configure the TCP/IP subsystem for the LIFs added in step 6, refer to the *TCP/IP (Parallel Library) Configuration and Management Manual* for Parallel Library TCP/IP (supported on NonStop S-series systems only), the *TCP/IP Configuration and Management Manual* for conventional TCP/IP, or the *TCP/IPv6 Configuration and Management Manual* for NonStop TCP/IPv6.

---

9. Use the SCF STATUS commands to check that the adapter, SAC(s), PIFs, and LIFs have started.

```
STATUS ADAPTER $ZZLAN.G11021

SLSA Status ADAPTER

Name                      State
$ZZLAN.G11021             STARTED

STATUS SAC $ZZLAN.G11021.*

SLSA Status SAC

Name                      Owner    State
$ZZLAN.G11021.0          0       STARTED

STATUS PIF $ZZLAN.G11021.*

SLSA Status PIF

Name                      State
$ZZLAN.G11021.0.A        STARTED
$ZZLAN.G11021.0.B        STARTED
$ZZLAN.G11021.0.C        STARTED
$ZZLAN.G11021.0.D        STARTED

STATUS LIF $ZZLAN.L111*

SLSA Status LIF

Name                      State      Access State
$ZZLAN.L11021A            STARTED   UP
$ZZLAN.L11021A            STARTED   UP
$ZZLAN.L11021A            STARTED   UP
$ZZLAN.L11021A            STARTED   UP
```

## Stopping and Starting an Adapter

When performing maintenance of the SLSA subsystem, you may need to stop or start an adapter. This subsection describes the procedures to use.

### Stopping an Adapter

Before you delete a ServerNet adapter, you must stop the adapter and its subordinate SAC(s) and PIFs on the adapter.

Perform the following steps to stop an adapter:

1. Determine the name of the adapter associated with the group and slot that the adapter is in. Use the SCF INFO ADAPTER command:

```
INFO ADAPTER $ZZLAN.*
```

```
SLSA INFO ADAPTER
```

Name	Group	Module	Slot	Type
\$ZZLAN.G11021	110	2	1	G4SA
\$ZZLAN.G11022	110	2	2	G4SA
\$ZZLAN.G11131	111	3	1	G4SA
\$ZZLAN.G11132	111	3	2	G4SA

2. Scan the list for the group and slot containing the adapter. For example, IOAM group 110, module 2, slot 1 contains the adapter named G11021.
3. Use the SCF INFO LIF command to list the LIF names in the system and identify the LIF names associated with the adapter. For example, LIFs L11021A, L11021B, L11021C, and L11021D are associated with adapter G11021.

```
INFO LIF $ZZLAN.*
```

```
SLSA Info LIF
```

Name	Associated Object	MAC Address	Type
\$ZZLAN.L11021A	G11021.0.A	08:00:8E:00:78:3A	Ethernet
\$ZZLAN.L11021B	G11021.0.B	08:00:8E:00:78:2D	Ethernet
\$ZZLAN.L11021C	G11021.1.C	08:00:8E:00:78:2C	Ethernet
\$ZZLAN.L11021D	G11021.1.D	08:00:8E:00:78:1C	Ethernet
\$ZZLAN.L11131A	G11131.0.A	08:00:8E:00:78:3A	Ethernet
\$ZZLAN.L11131B	G11131.0.B	08:00:8E:00:78:2D	Ethernet
\$ZZLAN.L11131C	G11131.1.C	08:00:8E:00:78:2C	Ethernet
\$ZZLAN.L11131D	G11131.1.D	08:00:8E:00:78:1C	Ethernet
\$ZZLAN.LANY	M0IE0.0.A	08:00:8E:00:7A:D9	Ethernet
\$ZZLAN.LANX	M0IE1.0.A	08:00:8E:00:7B:BA	Ethernet

4. Use the SCF STOP LIF command to stop the LIFs identified in step 3. If you cannot stop a LIF because it has active clients, use the SCF ABORT LIF command. You can use the SCF STATUS LIF command to verify that the LIFs are STOPPED.

```
STOP LIF $ZZLAN.L11021A
STOP LIF $ZZLAN.L11021B
STOP LIF $ZZLAN.L11021C
STOP LIF $ZZLAN.L11021D
STATUS LIF $ZZLAN.L11*
```

SLISA Status LIF

Name	State	Access State
\$ZZLAN.L11021A	STOPPED	UP
\$ZZLAN.L11021B	STOPPED	UP
\$ZZLAN.L11021C	STOPPED	UP
\$ZZLAN.L11021D	STOPPED	UP
\$ZZLAN.L11131A	STARTED	UP
\$ZZLAN.L11131B	STARTED	UP
\$ZZLAN.L11131C	STARTED	UP
\$ZZLAN.L11131D	STARTED	UP

5. Check the status of the PIFs and SAC(s) subordinate to the adapter. Use the SCF STATUS PIF and SCF STATUS SAC commands.

```
STATUS SAC $ZZLAN.G11021.*
```

SLISA Status SAC

Name	Owner	State
\$ZZLAN.G11021.0	0	STARTED

```
STATUS PIF $ZZLAN.G11021.*
```

SLISA Status PIF

Name	State	Trace Status
\$ZZLAN.G11021.0.A	STARTED	OFF
\$ZZLAN.G11021.0.B	STARTED	OFF
\$ZZLAN.G11021.0.C	STARTED	OFF
\$ZZLAN.G11021.0.D	STARTED	OFF

6. Stop the adapter and its subordinate SACs and PIFs by using the SCF STOP ADAPTER command with the SUB ALL option. If a subordinate object will not stop, use the SCF ABORT command for that object.

```
STOP ADAPTER $ZZLAN.G11021, SUB ALL
```

7. Use the SCF STATUS PIF and SCF STATUS SAC commands to check the status of the PIFs and SACs.

```
STATUS PIF $ZZLAN.G11021.*

SLSA Status PIF

Name                               State
$ZZLAN.G11021.0.A                 STOPPED
$ZZLAN.G11021.0.B                 STOPPED
$ZZLAN.G11021.0.C                 STOPPED
$ZZLAN.G11021.0.D                 STOPPED

STATUS SAC $ZZLAN.G11021*

SLSA Status SAC

Name                               Owner                               State
$ZZLAN.G11021.0                   0                               STOPPED
```

8. Verify that the adapter is in the STOPPED state by using the SCF STATUS ADAPTER command.

```
STATUS ADAPTER $ZZLAN.G11021

SLSA Status ADAPTER

Name                               State
$ZZLAN.G11021                     STOPPED
```

9. Perform any management tasks, such as removing, then installing a ServerNet adapter.

## Starting an Adapter

Perform the following steps to start an adapter you have added or replaced:

1. Start the adapter and its associated SAC(s) and PIFs by using the SCF START ADAPTER command, with the SUB ALL option. (This adapter might be the same one you stopped in step 7, under [Stopping an Adapter.](#))

```
START ADAPTER $ZZLAN.G11021, SUB ALL
```

2. After starting the adapter, use the SCF STATUS ADAPTER command to verify that the adapter was started.

```
STATUS ADAPTER $ZZLAN.G11021

SLSA Status ADAPTER

Name                               State
$ZZLAN.G11021                     STARTED
```



3. Verify that the SACs and PIFs are in the STARTED state by using the SCF STATUS SAC and SCF STATUS PIF commands.

```
STATUS SAC $ZZLAN.G11021*

SLSA Status SAC

Name                Owner      State
$ZZLAN.G11021.0      0        STARTED

STATUS PIF $ZZLAN.G11021*

SLSA Status PIF

Name                State
$ZZLAN.G11021.0.A    STARTED
$ZZLAN.G11021.0.B    STARTED
$ZZLAN.G11021.0.C    STARTED
$ZZLAN.G11021.0.D    STARTED
```

4. Start the LIFs associated with the PIFs on the adapter.

```
START LIF $ZZLAN.L11021A
START LIF $ZZLAN.L11021B
START LIF $ZZLAN.L11021C
START LIF $ZZLAN.L11021D
```

5. Verify the status of the LIFs by using the SCF STATUS LIF command.

```
STATUS LIF $ZZLAN.L11*

SLSA Status LIF

Name                State                Access State
$ZZLAN.L11021A      STARTED                UP
$ZZLAN.L11021B      STARTED                UP
$ZZLAN.L11021C      STARTED                UP
$ZZLAN.L11021D      STARTED                UP
$ZZLAN.L11131A      STARTED                UP
$ZZLAN.L11131B      STARTED                UP
$ZZLAN.L11131C      STARTED                UP
$ZZLAN.L11131D      STARTED                UP
```

## Renaming an Adapter

To rename an adapter, follow these steps.

1. Use the SCF STOP LIF or SCF ABORT LIF commands to stop the LIFs associated with the PIFs on the adapter you are renaming.

```
STOP LIF ($ZZLAN.L11021A, $ZZLAN.L11021B, $ZZLAN.L11021C,
$ZZLAN.L11021D)

SLSA E00012 Object $ZZLAN.L11021C has registered client(s) and
so cannot stop.

ABORT LIF $ZZLAN.L11021C
```

2. Stop the ADAPTER as described under [Stopping and Starting an Adapter](#).

```
STOP ADAPTER $ZZLAN.G11021, SUB ALL
```

The system stops the adapter and its subordinate SACs and PIFs.

3. Delete the LIFs you just stopped or aborted.

```
DELETE LIF ($ZZLAN.L11021A, $ZZLAN.L11021B, $ZZLAN.L11021C,  
$ZZLAN.L11021D)
```

4. Delete the name of the adapter with the SCF DELETE ADAPTER command.

```
DELETE ADAPTER $ZZLAN.G11021
```

5. Follow the procedure described under [Adding an Adapter to the System](#), to add the adapter (with a different name) back to the SLSA subsystem along with its associated LIFs.

```
ADD ADAPTER $ZZLAN.G4SA1, TYPE G4SA, LOCATION (110,2,1), &  
ACCESSLIST (0, 1, 2, 3)
```

```
START ADAPTER $ZZLAN.G4SA1, SUB ALL  
ADD LIF $ZZLAN.L11021A, PIF G4SA1.0.A  
ADD LIF $ZZLAN.L11021B, PIF G4SA1.0.B  
ADD LIF $ZZLAN.L11021C, PIF G4SA1.0.C  
ADD LIF $ZZLAN.L11021D, PIF G4SA1.0.D  
START LIF $ZZLAN.L11021A  
START LIF $ZZLAN.L11021B  
START LIF $ZZLAN.L11021C  
START LIF $ZZLAN.L11021D
```

## Altering the Access List

You may need to alter the access list that defines the data paths to ServerNet adapters connected to the server. You can change the order of the processors in the access list without stopping the LIFs or SACs but you must stop the LIFs or SACs before adding processors to or removing processors from the access list.

Follow these steps to change the data paths for a SAC:

1. Identify the name of the SAC for the ServerNet adapter you want to alter. For example, to alter the data path to the SAC on the adapter named G11021, you would alter the SAC named G11021.0.

2. Use the SCF INFO SAC command to check the current access list for the SAC you are planning to alter.

```
INFO SAC $ZZLAN.G11021.0

SLSA Info SAC

Name                Owner  *Access List
$ZZLAN.G11021.0    0      (0, 1, 2, 3)
```

3. Use the SCF ALTER SAC command to alter the access list.

```
ALTER SAC $ZZLAN.G11021.0, ACCESSLIST (1, 0, 2, 3)
```

The LANMAN process updates the access list for the specified SAC.

4. Use the SCF INFO SAC command to check the access list for the SAC you just altered. The system displays the information about the SAC.

```
INFO SAC $ZZLAN.G11021.0

SLSA Info SAC

Name                Owner  *Access List
$ZZLAN.G11021.0    1      (1, 0, 2, 3)
```

## Altering the LANMAN Process

You can alter the attributes of a LANMAN process through the NonStop Kernel subsystem manager. (Refer to the *SCF Reference Manual for the Kernel Subsystem*.) The LANMAN process can only run between processors 0 and 1 (same as \$SYSTEM).

## Aborting the LANMAN Process

When installing a new LANMAN process without cold loading the system, you must first abort the LANMAN process. Do not leave the LANMAN process \$ZZLAN in the aborted state any longer than necessary. (For more information about these commands, see the *SCF Reference Manual for the Kernel Subsystem*.)

---

**Note.** By default the `STOPMODE` should be set to `sysmsg`. To check the `STOPMODE` enter the `SCF INFO PROC $ZZKRN.#ZZLAN, detail` command.

---

If the `STOPMODE` is not `sysmsg`, follow these steps to abort the \$ZZLAN process:

1. Before you can abort the \$ZZLAN process, set the stopmode attribute of \$ZZKRN.#ZZLAN to `sysmsg`.
  - a. Issue the SCF ABORT PROC command.

```
ABORT PROC $ZZKRN.#ZZLAN
```

- b. Use the SCF ALTER PROC command to set the stopmode attribute

```
ALTER PROC $ZZKRN.#ZZLAN, stopmode sysmsg
```

2. Stop the persistence manager.

```
ABORT PROC $ZPM
```

3. Restart the Kernel process.

```
START PROC $ZZKRN.#ZZLAN
```

4. Use the SCF ABORT PROC command to abort the \$ZZLAN process.

```
ABORT PROC $ZZKRN.#ZZLAN
```

## Replacing a LANMON Without a Cold Load

As of the G06.21 RVU, a new LANMON can be installed without doing a cold load if your system has the LANMON SPR's requisites (see the LANMON SPR associated with your RVU for these requisites). If you do not have these requisites, you must install them and cold load the system.

Assuming the primary LANMAN is running in CPU 0, perform the following steps to replace a LANMON in all CPUs:

1. Follow the installation instructions described in the LANMON SPR associated with your RVU, including the DSM/SCM Build/Apply and ZPHIRNM Rename steps. However, do not cold load the system.

---

**Note.** When you restart the LANMON, the corrected LANMON object file that you installed in the *SYSn* subvolume will be used to start the LANMON process in that CPU.

---

2. Use the SCF ABORT MON command to ABORT and start the LANMONs in CPUs 02 through 15, sequentially, as follows:
  - a. Stop the LANMON in CPU 02.
  - b. Start the LANMON in CPU 02. Verify the LANMON has started.
  - c. Ensure that SAC, PIF, and LIF access is gained on CPU 02 before stopping the next LANMON.
  - d. Repeat substeps a, b, and c for each remaining CPU (that is, CPU 03 through CPU 15).

The following example stops and starts the LANMON in CPU 02 and verifies that the LANMON has started along with its associated SAC, PIF, and LIF objects:

```

ABORT MON $ZZLAN.#ZLM02
Status MON #ZLM02

Name           State      PID      Priority   Trace Status
$ZZLAN.#ZLM02  STOPPED
OFF

START MON $ZZLAN.#ZLM02

STATUS MON #ZLM02

SLSA Status MON
Name           State      PID      Priority   Trace Status
$ZZLAN.#ZLM02  STARTED   (3,302)   200       OFF

status sac G11021.0.A, detail

SLSA Detailed Status SAC \SYS.$ZZLAN.G11021.0.A

Current Access..... ( 0, 2, 3 )
Last Download Time..... 20 May 2003, 15:12:53.185
Last Error..... (3, LANMON, 2015)
Owner CPU..... 0
State..... STARTED
Trace Filename.....
Trace Status..... OFF

Fabric Status

CPU Accesslist   Fabric-X   Fabric-Y
-----
0                UP-PRIMARY UP
1                UP-PRIMARY UP
2                UP-PRIMARY UP
3                UP-PRIMARY UP

STATUS PIF $ZZLAN.G11021.0.A , DETAIL

SLSA Detailed Status PIF \SYS.$ZZLAN.G11021.0.A

CPUs with Data Path..... ( 2 )
Last Error..... (0,0,0)
State..... STARTED
Trace Filename.....
Trace Status..... OFF

E4SA Controller Status

Link Pulse State..... UP

STATUS LIF $ZZLAN.L11*

SLSA Status LIF

Name           State      Access State
$ZZLAN.L11021A  STARTED   UP
$ZZLAN.L11021B  STARTED   UP
$ZZLAN.L11021C  STARTED   UP
$ZZLAN.L11021D  STARTED   UP

```

- Determine if the primary LANMAN is in CPU 0 or CPU 01 by issuing the SCF STATUS PROCESS command:

```
STATUS PROCESS $ZZKRN.#ZZLAN

NONSTOP KERNEL - Status PROCESS \HELLO.$ZZKRN.#ZZLAN
Symbolic Name      Name      State      Sub Primary Backup      Owner
                                PID      PID
ID
$ZZLAN              $ZZLAN    STARTED      (0 ,15)    (1 ,16)
255,255
NON STOP KERNEL
SLSA Status PROCESS
```

- Use the SCF ABORT MON command to ABORT the LANMON in the CPU associated with the backup LANMAN. For example, if the backup LANMAN is in CPU 01:

```
ABORT MON $ZZLAN.#ZLM01
```

- Use the SCF START MON command to start the LANMON in CPU 01 (if the backup LANMAN is in CPU 01). For example:

```
START MON $ZZLAN.#ZLM01
```

6. Verify that the LANMON in CPU 01 has started and that SAC, PIF, and LIF access is gained on that CPU:

```

Status MON #ZLM01

Name                State      PID      Priority  Trace Status
$ZZLAN.#ZLM01      STARTED   (3,302)    200      OFF

status sac G11021.0.A, detail

SLSA Detailed Status SAC \SYS.$ZZLAN.G11021.0.A

Current Access..... ( 0, 2, 3 )
Last Download Time..... 20 May 2003, 15:12:53.185
Last Error..... (3, LANMON, 2015)
Owner CPU..... 0
State..... STARTED
Trace Filename.....
Trace Status..... OFF

Fabric Status

  CPU Accesslist   Fabric-X   Fabric-Y
  -----
      0           UP-PRIMARY   UP
      1           UP-PRIMARY   UP
      2           UP-PRIMARY   UP
      3           UP-PRIMARY   UP

STATUS PIF $ZZLAN.G11021.0.A , DETAIL

SLSA Detailed Status PIF \SYS.$ZZLAN.G11021.0.A

CPUs with Data Path..... ( 1 )
Last Error..... (0, 0, 0)
State..... STARTED
Trace Filename.....
Trace Status..... OFF

E4SA Controller Status

Link Pulse State..... UP

STATUS LIF $ZZLAN.L11*

SLSA Status LIF

Name                State      Access State
$ZZLAN.L11021A      STARTED   UP
$ZZLAN.L11021B      STARTED   UP
$ZZLAN.L11021C      STARTED   UP
$ZZLAN.L11021D      STARTED   UP

```

7. Use the SCF SWITCH PROCESS command to switch operation of the LANMON process to the backup LANMON process.

```
SWITCH PROCESS $ZZLAN
```

8. Verify that the primary LANMAN has switched CPUs by issuing the SCF STATUS PROCESS command:

```
STATUS PROCESS $ZZKRN.#ZZLAN

NONSTOP KERNEL - Status PROCESS \HELLO.$ZZKRN.#ZZLAN

Symbolic Name          Name    State   Sub Primary   Backup   Owner
                        PID      PID      PID      PID      ID
$ZZLAN                 $ZZLAN STARTED    1 ,15    0 ,16    255,255
```

9. Use the SCF ABORT MON command to ABORT the LANMON in CPU 0 (if your backup LANMAN is in CPU 0). If your backup LANMAN is in CPU 01, abort the LANMON in CPU 01.

```
ABORT MON $ZZLAN.#ZLM00
```

10. Use the SCF START MON command to start the LANMON in CPU 0 (if your backup LANMAN is in CPU 0). If your backup LANMAN is in CPU 01, start the LANMON in CPU 01.

```
START MON $ZZLAN.#ZLM00
```

## Considerations

- Starting \$ZZKRN.#ZZLAN causes all LANMONs to be created and started automatically. The fact that a LANMON was previously aborted is lost when \$ZZLAN is aborted.
- The system will need to be cold loaded the first time the new software is installed.
- The amount of time a LANMON is aborted should be kept to a minimum.
- A client may be prevented from switching if the LANMON in which the client's backup is running has been aborted.



# **A** Command Summary

The appendix shows the syntax of the SCF commands for the ServerNet LAN Systems Access (SLSA) subsystem.

```
ABORT [ /OUT file-spec/ ] ADAPTER adapter-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

```
ABORT [ /OUT file-spec/ ] LIF lif-name
```

```
ABORT [ /OUT file-spec / ] MON lanmon-name
```

```
ABORT [ /OUT file-spec/ ] PIF pif-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

```
ABORT [ /OUT file-spec/ ] SAC sac-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

```
ADD [ /OUT file-spec/ ] ATMSAP atmsap-name  
    [ , MTU max-mtu-size ]  
    [ , NATURE { PVC } ]  
    [ , QOSSET { UBR } ]  
    [ , VCC ( vpi, vci ) ]
```

```
ADD [ /OUT file-spec/ ] ADAPTER adapter-name

{ , TYPE {E4SA|MIOE|TRSA|FESA|GESA|G4SA|ATM3SA|CCSA}
  , LOCATION (group,module,slot)
  , ACCESSLIST ( n0 , n1,..., n15 )
[ , AUTODUMP { ON | OFF |EXTENDED}]
[ , AUTOFIRMUP { ON | OFF }
[ , AUTOSTART { ON | OFF }
[ , DLFILENAME file-spec
[ , DUMPFILNAME file-spec
[ , FIRMWAREFILENAME file-spec ]
```

```
ADD [ /OUT file-spec/ ] LIF lif-name , PIF pif-name |
ATMSAP atmsap.name

[ , DATAFORWARDMODE df-mode ]
[ , DATAFORWARDUNIT df-unit ]
[ , DATAFORWARDCOUNT df-count ]
[ , DATAFORWARDTIME df-time ]
```

```
ALTER [ /OUT file-spec/ ] ATMSAP atmsap-name
[ , MTU max-mtu-size ]
[ , NATURE { PVC } ]
[ , QOSSET { UBR } ]
[ , VCC ( vpi, vci )]
```

```
ALTER [ /OUT file-spec/ ] LIF lif-name
[ , DATAFORWARDMODE df-mode ]
[ , DATAFORWARDUNIT df-unit ]
[ , DATAFORWARDCOUNT df-count ]
[ , DATAFORWARDTIME df-time ]
```

```
ALTER [ /OUT file-spec/ ] PIF pif-name [ , RESET ]

    [ , ACTIVEMONITOR { ON | OFF } ]
    [ , AUTONEGOTIATION { ON | OFF } ]
    [ , INTERFACE { AUTODETECT | COPPER | FIBER } ]
    [ , DATAFORWARDCOUNT df-count ]
    [ , DATAFORWARDTIME df-time ]
    [ , DUPLEX { HALF | FULL } ]
    [ , EARLYTOKENRELEASE { ON | OFF } ]
    [ , EMSVERBOSE { ON | OFF } ]
    [ , JUMBOFRAME { ON | OFF } ]
    [ , LINESPEED { 10 | 100 | 1000 } ]
    [ , MAXSESSIONS max-sessions ]
    [ , NODEMACADDRESS { addr | DEFAULT } ]
    [ , NOOPTION ]
    [ , RINGSPEED { 4 | 16 } ]
```

```
ALTER [ /OUT file-spec/ ] SAC sac-name

    { [ , ACCESSLIST ( n0 , n1 , ... , n15 ) ]
      [ , AUTODUMP { ON | OFF | EXTENDED } ]
      [ , AUTOFIRMUP { ON | OFF } ]
      [ , AUTOSTART { ON | OFF } ]
      [ , DLFILENAME file-spec ]
      [ , FIRMWAREFILENAME file-spec ]
      [ , DUMPFILENAME file-spec ] }
```

```
DELETE [ /OUT file-spec/ ] ADAPTER adapter-name
```

```
DELETE [ /OUT file-spec/ ] LIF lif-name
```

```
INFO [ /OUT file-spec/ ] ADAPTER adapter-name

    [ , DETAIL | OBEYFORM ]
```

```
INFO [ /OUT file-spec/ ] ATMSAP atmsap-name

    [ , {DETAIL | OBEYFORM} ]
```

```
INFO [ /OUT file-spec/ ] LIF lif-name [ , DETAIL | OBEYFORM]
```

```
INFO [ /OUT file-spec/ ] MON lanmon-name [ , DETAIL ]
```

```
INFO [ /OUT file-spec/ ] PIF pif-name [ , DETAIL | OBEYFORM]
```

```
INFO [ /OUT file-spec/ ] SAC sac-name [ , DETAIL | OBEYFORM]
```

```
LISTOPENS LIF [/OUT file-spec / ] LIF [ lif-name ]
```

```
NAMES [ /OUT file-spec / ][ object-name ]
```

```
NAMES [ /out-file-spec / ] ATMSAP atmsap-name
```

```
NAMES [ /OUT file-spec / ] ADAPTER adapter-name
```

```
NAMES [ /OUT file-spec / ] LIF lif-name
```

```
NAMES [ /OUT file-spec / ] MON lanmon-name
```

```
NAMES [ /OUT file-spec / ] PIF pif-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

```
NAMES [ /OUT file-spec / ] PROCESS $process-name
```

```
NAMES [ /OUT file-spec / ] SAC sac-name
```

```
RESET [ /OUT file-spec / ] ADAPTER adapter-name
```

```
START [ /OUT file-spec/ ] ATMSAP atmsap-name
```

```
START [ /OUT file-spec / ] ADAPTER adapter-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

```
START [ /OUT file-spec / ] LIF lif-name
```

```
START [ /OUT file-spec / ] MON lanmon-name
```

```
START [/OUT file-spec/ ] PIF pif-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

```
START [ /OUT file-spec/ ] SAC sac-name  
      [ , SUB [ ONLY | ALL | NONE ] ]
```

```
STATS [ /OUT file-spec/ ] ATMSAP atmsap-name [ ,RESET]
```

```
STATS [ /OUT file-spec / ] PIF pif-name [ , RESET ]
```

```
STATUS [ /OUT file-spec/ ] ATMSAP atmsap-name [ , DETAIL ]
```

```
STATUS [ /OUT file-spec/ ] ADAPTER adapter-name [ , DETAIL ]
```

```
STATUS [ /OUT file-spec/ ] LIF lif-name [ , DETAIL ]
```

```
STATUS [ /OUT file-spec/ ] MON lanmon-name [ , DETAIL ]
```

```
STATUS [ /OUT file-spec/ ] PIF pif-name [ , DETAIL ]
```

```
STATUS [ /OUT file-spec/ ] PROCESS $ZZLAN [ , DETAIL ]
```

```
STATUS [ /OUT file-spec/ ] SAC sac-name [ , DETAIL ]
```

```
STOP [ /OUT file-spec/ ] ATMSAP atmsap-name
```

```
STOP [ /OUT file-spec/ ] ADAPTER adapter-name  
[ , SUB [ ONLY | ALL | NONE ] ]
```

```
STOP [ /OUT file-spec/ ] LIF lif-name
```

```
STOP [ /OUT file-spec/ ] PIF pif-name
      [ , SUB [ ONLY | ALL | NONE ] ]
```

```
STOP [ /OUT file-spec/ ] SAC sac-name
      [ , SUB [ ONLY | ALL | NONE ] ]
```

```
SWITCH [ /OUT file-spec/ ] PROCESS lanman-name
```

```
TRACE [ /OUT file-spec/ ] MON lanmon-name
      { , STOP
        [ , BULKIO | NOBULKIO
        [ , COUNT count
        [ , LOCKSIZE locksize
        [ , NOCOLL
        [ , PAGES pages
        [ , RECSIZE size
        [ , SELECT select-spec
        [ , TO file-spec
        [ , WRAP
      }
```

```
TRACE [ /OUT file-spec/ ] PIF pif-name
      { , STOP
      { [ , BULKIO | NOBULKIO
        [ , COUNT count
        [ , CPU cpu-number
        [ , LOCKSIZE locksize
        [ , NOCOLL
        [ , PAGES pages
        [ , RECSIZE size
        [ , SELECT select-spec
        [ , TO file-spec
        [ , WRAP
      }
```

```
TRACE [ /OUT file-spec/ ] PROCESS process-name
```

```

{ , STOP }
{ , [ , BULKIO | NOBULKIO ]
  [ , COUNT count ]
  [ , LOCKSIZE locksize ]
  [ , NOCOLL ]
  [ , PAGES pages ]
  [ , RECSIZE size ]
  [ , SELECT select-spec ]
  [ , TO file-spec ]
  [ , WRAP ] }
```

```
TRACE [ /OUT file-spec/ ] SAC sac-name
```

```

{ , STOP }
{ , [ , BULKIO | NOBULKIO ]
  [ , COUNT count ]
  [ , CPU cpu-number ]
  [ , LOCKSIZE locksize ]
  [ , NOCOLL ]
  [ , PAGES pages ]
  [ , RECSIZE size ]
  [ , SELECT select-spec ]
  [ , TO file-spec ]
  [ , WRAP ] }
```

```
VERSION [/OUT file-spec/ ] [ object-name ] [ , DETAIL ]
```

```
VERSION [ /OUT file-spec/ ] MON lanmon-name [ , DETAIL ]
```

```
VERSION [ /OUT file-spec/ ] PROCESS process-name, [DETAIL]
```



# B SCF Error Messages

This appendix lists the error messages that SCF can return for the SLSA subsystem and describes the cause, effect, and recovery for each error.

## SLSA 00001

SLSA E00001 Command not yet implemented.

**Cause.** You entered a command that SCF does not recognize.

**Effect.** SCF ignores the command.

**Recovery.** Enter the appropriate command.

## SLSA 00002

SLSA E00002 SLSA SCF Product Module internal error - contact TNSC.

**Cause.** SCF encountered an internal error.

**Effect.** SCF ignores the command.

**Recovery.** This is a serious error. Contact your service provider.

## SLSA 00003

SLSA E00003 Duplicate attribute specified.

**Cause.** You specified an attribute more than once in a single command.

**Effect.** SCF ignores the command.

**Recovery.** Retry the command with the appropriate number of attributes.

## SLSA 00004

SLSA E00004 SLSA SCF Product Module Internal Error: Case value out of range

**Cause.** An invalid case value was generated with no associated case label.

**Effect.** SCF ignores the command.

**Recovery.** This is a serious error. Contact your service provider.

## SLSA 00005

SLSA E00005 Object *object-name* has been given an invalid access list.

**Cause.** An invalid access list (such as, unacceptable processor, nonexistent processor, or incorrect number of processors) has been specified in the ALTER SAC command.

**Effect.** SCF ignores the command, and the access list is not altered for the specified SAC.

**Recovery.** Check the command, then reissue it. The access list must have from 2 to 16 valid processors specified for a SAC. For a MIOE adapter, the access list cannot exceed 2 processors.

## SLSA 00006

SLSA E00006 Object name *object-id* contains wild cards - not allowed in object name for this command.

**Cause.** You attempted to use wild-card characters (\*, ?) in an object name for a command that does not support wild-cards.

**Effect.** SCF ignores the command.

**Recovery.** Reissue the command without wild-card characters in the object name.

## SLSA 00007

SLSA E00007 Internal Error *error-number*, Origin *origin*, Severity level for *object-name*.

**Cause.** An internal error occurred within the SLSA subsystem (LANMAN, LANMON, DIH, QIO, XIO, ServerNet).

**Effect.** SCF ignores the command.

**Recovery.** This is a serious error. Contact your service provider with the *error-number*, *origin*, *level*, and *object-name* information.

*error-number*

specifies an internal error number.

*origin*

specifies where the internal error originated, such as LANMAN, LANMON, DIH, QIO, XIO, SvNET

*level*

specifies whether this error is informative, warning, fatal, or unrecognized.

*object-name*

specifies the name of the object that generated the internal error.

## SLSA 00008

SLSA E00008 Configuration failure by *origin* due to *cause* (*detail*) on operation *operation-type* for *object-name*.

**Cause.** The SCF command issued to the object specified by *object-name* requires LANMAN to manipulate the configuration database. The manipulation failed as specified by the *cause* and *detail* information.

**Effect.** SCF ignores the command.

**Recovery.** Check the command and correct all problems, then try the command again. If failure persists, contact your service provider with the *origin*, *cause*, and operation information.

*origin*

specifies the part of the subsystem where the error originated.

*cause (detail)*

specifies what caused the error. The errors are:

- Invalid group, module, slot
- Invalid slot number
- Invalid adapter type
- Record not found
- Record found
- Location already used
- Obsolete record version

*operation-type*

specifies the operation the object was performing when the error occurred. The operations are:

- Locking
- Unlocking
- Inserting
- Deleting
- Reading
- Updating

*object-name*

specifies the name of the object that generated the error.

## SLSA 00009

```
SLSA E00009 Object object-name is busy with other operations.
```

**Cause.** The object you issued the command to is busy performing actions and cannot process the SCF command.

**Effect.** SCF ignores the command.

**Recovery.** Wait, then try the command again. Contact your service provider if the problem persists for a long time.

## SLSA 00010

```
SLSA E00010 Cannot start object-name. Parent object is  
still in STOPPED state.
```

**Cause.** The SLSA subsystem received a START command for an object (specified by *object-name*) whose parent is still in the STOPPED state. The object cannot be started until the parent object is STARTED.

**Effect.** The specified object is not started.

**Recovery.** Start the parent object, then reissue the command.

## SLSA 00011

```
SLSA E00011 Cannot stop object-name because subordinate  
object is not stopped.
```

**Cause.** A STOP or an ABORT command was received on an object that has one or more subordinate objects not in the STOPPED state. Objects cannot be put in the STOPPED state unless all subordinate objects are in the STOPPED state.

**Effect.** STOP or ABORT command has no effect on the state of the specified object.

**Recovery.** Check status of subordinate objects, and put all of them in the STOPPED state, then reissue the command.

## SLSA 00012

SLSA E00012 Object *object-name* has registered client(s) and cannot stop.

**Cause.** A LIF is preventing a LIF or a PIF from being stopped because the LIF and its associated PIF have active clients. *object-name* specifies the LIF or a PIF that has the registered clients.

**Effect.** You cannot stop the object specified by *object-name*.

**Recovery.** Identify the LIF involved (use the INFO LIF command), terminate all registered clients of the LIF, then reissue the STOP command. Use the ABORT command instead of the STOP command if you still cannot stop the object.

## SLSA 00013

SLSA E00013 Command for object *object-name* failed due file system error *error-number*.

**Cause.** A LOAD or DUMP command issued by the system console failed during a file-system operation on the firmware or dump file specified in the command. This action caused the command to fail.

**Effect.** The firmware is not downloaded or the dump file is not generated.

**Recovery.** Make sure that the SAC object has a valid and correct firmware or dump file name (use the INFO SAC or ALTER SAC command), then reissue the command.

## SLSA 00014

SLSA E00014 Command failed for object *object-name* due to QIO error *error-number*.

**Cause.** The specified SCF command required use of QIO resources, but an error in manipulating these QIO resources caused the command to fail.

**Effect.** SCF ignores the command.

**Recovery.** Check the condition that caused the given QIO error, resolve that condition, then retry the SCF command.

## SLSA 00015

SLSA E00015 PIF specified for *lif-name* is already in use by another LIF.

**Cause.** An ADD LIF command was issued to a PIF that is already in use by a LIF. A PIF can be used by only one LIF.

**Effect.** The LIF is not assigned to the specified PIF.

**Recovery.** Assign a different PIF to the LIF you are attempting to add, or remove the current PIF from the LIF you specified in the ADD LIF command. Use the INFO LIF command to display the PIFs assigned to the LIFs in the SLSA subsystem.

## SLSA 00016

```
SLSA E00016 Subordinate PIF of object-name has link to  
existing LIF object.
```

**Cause.** A DELETE ADAPTER command was issued for an ADAPTER object that has one or more subordinated PIFs linked to existing LIF objects. You cannot delete an ADAPTER object until there are no subordinate objects (LIFs) dependent on PIFs of the specified adapter.

**Effect.** You cannot delete the adapter object.

**Recovery.** To remove the ADAPTER object, you must first stop all LIFs that are linked to PIFs subordinated to the specified adapter. Use the INFO LIF command to display the names of the LIFs and their associated PIFs.

## SLSA 00017

```
SLSA E00017 Command not supported by adapter type of object  
object-name.
```

**Cause.** A command was issued to an object (specified by *object-name*) whose parent is of an adapter type (for example, MFIOB) that does not support the specified command.

**Effect.** SCF ignores the command.

**Recovery.** Reissue the command using a supported adapter type (for example, E4SA).

## SLSA 00018

```
SLSA E00018 Content of firmware file for object-name does not  
have valid information.
```

**Cause.** A LOAD command was issued for a SAC object (specified by *object-name*) whose previously specified firmware file contains inappropriate information for a downloadable firmware file.

**Effect.** The firmware is not downloaded to the SAC.

**Recovery.** Use the INFO SAC and ALTER SAC commands to make sure the SAC has a valid firmware file specified for it, then reissue the command.

## SLSA 00019

SLSA E00019 Firmware file for *object-name* does not contain an applicable download module.

**Cause.** A LOAD command was issued for a SAC object, specified by *object-name*, whose previously specified firmware file does not contain a downloadable module that is appropriate for that particular SAC object.

**Effect.** The firmware file is not downloaded to the SAC.

**Recovery.** Make sure that the SAC object has a valid and correct firmware file (use the INFO SAC and ALTER SAC commands) then reissue the command.

## SLSA 00020

SLSA E00020 Invalid dump address information for object *object-name*.

**Cause.** A DUMP command was issued for a SAC object that is using invalid dump-address information.

Two possible causes of bad dump-address information are:

- The SAC never downloaded a valid application-download file because no such file exists or because the SAC was never started.
- The application-download file contains invalid dump-address information.

**Effect.** The dump is not performed.

**Recovery.** Make sure that a valid application-download file with valid dump address information exists, then make sure that the application-download file downloads to the adapter.

## SLSA 00021

SLSA E00021 Command not allowed while object in *state* state for *object-name*

**Cause.** A command was issued and the current summary state of the target object prevents processing of the command. This error is returned whenever the state-transition specification is not met.

**Effect.** SCF ignores the command.

**Recovery.** Ensure that the target object is in a summary state compatible with the command to be issued.

## SLSA 00022

```
SLSA E00022 filename is an invalid filename
```

**Cause.** The file name specified in an ADD ADAPTER or ALTER SAC command is not valid or does not contain enough information to construct a fully qualified file name.

**Effect.** SCF ignores the command.

**Recovery.** Check the file name, then reissue the command.

## SLSA 00023

```
SLSA E00023 filename is an invalid dump filename
```

**Cause.** The dump file name in an ADD ADAPTER or ALTER SAC command is not valid because it does not end with 00.

**Effect.** SCF ignores the command.

**Recovery.** Use a valid file name that ends with 00 , then reissue the command.

## SLSA 00024

```
SLSA E00024 SLSA process is busy - cannot process request
```

**Cause.** A SLSA process (either LANMAN or one of the LANMONs) is busy and cannot process the given request at this time. LANMAN or LANMON is probably starting up.

**Effect.** SCF ignores the command.

**Recovery.** Retry command later. If this problem persists, contact your service provider.

## SLSA 00025

```
SLSA E00025 SLSA command timed out on object-name
```

**Cause.** A command expired before a response from the adapter was received.

**Effect.** The SCF command you entered was initiated, but successful completion cannot be guaranteed.

**Recovery.** If possible, check if the command completed successfully; otherwise, reissue the command.



## SLSA 00026

SLSA E00026 Adapter corresponding to *object-name* is configured but not detected at present

**Cause.** The SLSA subsystem does not detect the adapter specified by *object-name* because the adapter is not plugged in, or it is the process of being reset.

**Effect.** SCF ignores the command.

**Recovery.** Retry the SCF command. If this message persists, make sure that adapter is correctly plugged in, then retry the command.

## SLSA 00027

SLSA E00027 Adapter type of *object-name* does not match actual physical adapter type

**Cause.** You specified an adapter type that does not match the type of adapter installed in the system at the specified location.

**Effect.** SCF ignores the command.

**Recovery.** Check that the right adapter is in the right location. Then reconfigure adapter type.

## SLSA 00029

SLSA E00029 Adapter type of *object-name* does not support changing of specified attribute(s)

**Cause.** One or more attributes that you specified in an ALTER command does not exist for the type of adapter you specified or cannot be changed for that adapter type. For example, for PIF attributes, RINGSPEED is valid only for adapter type TRSA and you can set NODEMACADDRESS only for adapter type TRSA. Attempting to set these attributes for a PIF on an E4SA adapter type would cause error 29.

**Effect.** SCF ignores the command.

**Recovery.** Check that the object specified by *object-name* is correct. If you specified the incorrect object name, reissue the command using the correct *object-name*. Otherwise, the command is not supported for the given object.

## SLSA 00030

```
SLSA E00030 Given MAC address is not valid as  
locally-administered MAC address
```

**Cause.** You specified a MAC address that is not between %H400000000000 and %H7FFFFFFFFFFFF. MAC addresses in this range are available as a locally-administered MAC address.

**Effect.** The MAC address is not changed, and the command does not complete.

**Recovery.** Reissue the command using a valid, locally-administered MAC address.

## SLSA 00046

```
SLSA E000 Subordinated PIF of <objname> has existing objects  
subordinated to it
```

**Cause.** A DELETE ADAPTER command was issued for an ADAPTER that has one or more subordinated PIFs, each of which, in turn, has one or more subordinate objects. The ADAPTER object cannot be removed while objects are subordinated to the PIFs

**Effect.** SCF ignores the command.

**Recovery.** Remove all objects that are subordinated, then issue the DELETE ADAPTER command.

## SLSA 00048

```
SLSA E00048 Maximum number of ATMSAP exceeded
```

**Cause.** You issued an ADD ATMSAP command, and tried to subordinate it to a PIF object which has 128 ATMSAP objects already subordinated to it.

**Effect.** SCF ignores the command.

**Recovery.** Up to 128 ATMSAP objects can be subordinated to a PIF on an ATM3SA adapter. Delete an existing ATMSAP before adding the new one, or assign the new ATMSAP to another PIF object that has less than 128 ATMSAPs subordinated to it.

## SLSA 00050

```
SLSA E00050 DATAFORWARDCOUNT and DATAFORWARDTIME incompatible  
with non-PIF LIF
```

**Cause.** You issued an ADD or ALTER LIF command in which the attributes DATAFORWARDCOUNT and DATAFORWARDTIME were specified, but the LIF was associated with a ATMSAP object.

**Effect.** SCF ignores the command.

**Recovery.** Remove the DATAFORWARDCOUNT and DATAFORWARDTIME attributes from the ADD or ALTER command line. These two attributes are not required for a LIF that is associated with an ATMSAP object.

## SLSA 00051

```
SLSA E00051 Superior object not started for <obj name>
```

**Cause.** You issued a STATUS command, however the status information is not available for display. The object or its parent object is not in a started state.

**Effect.** SCF ignores the command.

**Recovery.** Start the object and/or parent object before issuing the status command.

## SLSA 00052

```
SLSA E00052 Command not valid for object <obj name>.
```

**Cause.** You issued an SCF command, however the SCF command is not supported by that particular object.

**Effect.** SCF ignores the command.

**Recovery.** See [SLSA Subsystem SCF Objects](#) on page 2-11 for further information.

## SLSA 00056

```
SLSA E00056 Attribute value invalid: <obj name>.
```

**Cause.** You issued an ADD or ALTER command, and tried to assign an invalid value to an attribute.

**Effect.** SCF ignores the command.

**Recovery.** Set the attribute to a value that is within the valid range.

## SLSA 00060

```
SLSA E00060 LINESPEED and DUPLEX required when AUTONEGOTIATE  
is set to OFF
```

**Cause.** You issued an ALTER PIF command on a FESA or GESA adapter. You specified the AUTONEGOTIATE attribute to OFF, but the LINESPEED and DUPLEX attributes were left unspecified.

**Effect.** SCF ignores the command.

**Recovery.** Specify the LINESPEED and DUPLEX attribute settings in the ALTER PIF Command.

## SLSA 00061

```
SLSA E00061 LINESPEED and DUPLEX not required when  
AUTONEGOTIATE is set to ON
```

**Cause.** You issued an ALTER PIF command on a FESA or GESA adapter. You specified the AUTONEGOTIATE attribute to ON; however, you also specified the LINESPEED and DUPLEX attributes in the command line.

**Effect.** SCF ignores the command.

**Recovery.** Delete the LINESPEED and DUPLEX attribute settings in the ALTER PIF Command.

## SLSA 00062

```
SLSA E00062 CPU is not on the ACCESSLIST for objname
```

**Cause.** You issued a TRACE PIF or TRACE SAC command with the CPU attribute specifying a CPU number that was not on the ACCESSLIST of the PIF or the SAC.

**Effect.** SCF ignores the command.

**Recovery.** Specify a valid CPU number in the TRACE command.

## SLSA 00063

```
SLSA E00063 CPU attribute not supported for TRACE MON or  
TRACE PROCESS command
```

**Cause.** You issued a TRACE MON or PROCESS command with the CPU attribute specified.

**Effect.** SCF ignores the command.

**Recovery.** Remove the CPU attribute in the TRACE command.

## SLSA 00064

```
SLSA E00064 Invalid TRACE command attribute combination
```

**Cause.** The attribute in question conflicted with one or more of the other attributes on the command line.

**Effect.** SCF ignores the command.

**Recovery.** Refer to the [TRACE Command](#) on page 4-135 to determine the attributes that can be combined with a TRACE command string.

## SLSA 00065

```
SLSA E00065 LOCKSIZE must be less than or equal to PAGES
```

**Cause.** You issued a TRACE command with a LOCKSIZE value that was greater than the PAGES value.

**Effect.** SCF ignores the command.

**Recovery.** Retry the command with a LOCKSIZE value that is less than or equal to the PAGES value. If PAGES is not specified, LOCKSIZE must be less than or equal to 64 pages.

## SLSA 00066

```
SLSA E00066 Cannot stop objname because objname is running  
in the same CPU as the primary LANMAN
```

**Cause.** The user issued an ABORT MON command against the monitor that is running in the same CPU as the primary LANMAN. This action is not allowed because LANMAN needs a local LANMON to function correctly.

**Effect.** SCF ignores the command.

**Recovery.** Issue the SWITCH PROCESS command, then retry the ABORT MON command.

## SLSA 00067

```
SLSA E00067 Cannot switch $ZZLAN because the MON in the  
backup CPU is STOPPED
```

**Cause.** The user issued a SWITCH PROCESS command but the backup process has no associated monitor process. This action is not allowed because LANMAN needs a local LANMON to function correctly.

**Effect.** SCF ignores the command.

**Recovery.** Issue the START MON command for the LANMAN's backup CPU, then retry the SWITCH PROCESS command.

## SLSA 00068

```
SLSA E00068 The MON in the specified cpu to be traced is  
in the STOPPED state
```

**Cause.** The user issued a TRACE command but there is no LANMON to handle that request in the specified CPU.

**Effect.** SCF ignores the command.

**Recovery.** Issue the START MON command, then retry the TRACE command.

## SLSA 00071

```
SLSA E00071 The objname does not support the FIBER  
INTERFACE type.
```

**Cause.** The user attempted to specify an INTERFACE attribute value which is not supported by the specified physical interface (PIF). G4SA adapters have four configurable physical interfaces (PIFs). PIF A and PIF B can only be configured with INTERFACE values COPPER or AUTODETECT; however, PIF C and PIF D can be configured with INTERFACE values COPPER, FIBER, or AUTODETECT.

**Effect.** SCF ignores the command.

**Recovery.** Reissue the command with INTERFACE set to COPPER or AUTODETECT, or specify a different physical interface (PIF) which supports the specified INTERFACE value.

## SLSA 00072

```
SLSA E00072 DATAFORWARDTIME is beyond the valid range of  
1-200 MILLISECONDS
```

**Cause.** The user attempted to specify a DATAFORWARDTIME beyond the range of the current DATAFORWARDUNIT.

**Effect.** SCF ignores the command.

**Recovery.** Reissue the command with a DATAFORWARDTIME value that is between 1 and 200 milliseconds.

## SLSA 00073

```
SLSA E00073 SLSA E00073 DATAFORWARDTIME is beyond the  
valid range of 1-200000 MICROSECONDS
```

**Cause.** The user attempted to specify a DATAFORWARDTIME value beyond the range of the current DATAFORWARDUNIT.

**Effect.** SCF ignores the command.

**Recovery.** Reissue the command with a DATAFORWARDTIME value that is between 1 and 200000 MICROSECONDS.





# Index

## A

ABORT command

ADAPTER [4-7](#)

ATMSAP [4-9](#)

LIF [4-10](#), [4-130](#), [5-7](#)

MON [4-11](#)

PIF [4-12](#), [4-132](#)

SAC [4-133](#)

Access list [4-16](#), [4-36](#), [4-37](#), [4-62](#), [4-111](#)

altering [5-8](#)

ADAPTER

aborting [4-7](#)

adding [4-15](#)

deleting [4-39](#)

INFO command [4-41](#)

Naming [4-67](#)

RESET command [4-76](#)

starting [4-76](#)

STATUS command [4-106](#)

stopping [4-128](#)

subordinate objects [4-18](#)

Adapters

configuring using WAN Wizard Pro [1-8](#), [5-1](#)

display of names [4-74](#)

installing [5-2](#)

naming [5-1](#)

naming convention for [3-2](#)

ADD command

ADAPTER [4-15](#), [5-2](#)

ATMSAP [4-21](#)

LIF [4-23](#)

ALTER command

ATMSAP [4-27](#)

LIF [4-29](#)

PIF [4-32](#)

SAC [4-18](#), [4-36](#), [5-2](#), [5-9](#)

Application microcode [4-17](#), [4-62](#), [4-63](#), [4-127](#)

Asterisk [4-41](#)

ATMSAP [4-2](#), [4-4](#)

aborting [4-9](#)

adding [4-21](#)

altering [4-27](#)

deleting [4-39](#)

Expand [2-15](#)

INFO command [4-44](#)

NAMES command [4-67](#)

naming conventions [3-8](#)

object description [2-14](#)

starting [1-7](#), [4-77](#)

STATS command [4-83](#)

STATUS command [4-107](#)

stopping [4-129](#)

## B

Bulk I/O [4-135](#), [4-139](#), [4-142](#), [4-145](#)

## C

cabid [3-3](#)

CENTRY object [2-14](#)

Clients [2-9](#), [4-10](#), [4-13](#), [4-130](#), [4-132](#)

Cold-load

See system-load

Configuration

file [3-12](#)

TACL command file [3-12](#)

Configuration forms [5-1](#)

completing [3-9](#)

CRU [2-12](#)

See also PMF CRU

## D

Data path [4-110](#), [4-115](#), [4-116](#), [4-117](#), [4-118](#), [4-120](#), [4-126](#), [4-133](#)

DELETE command

ADAPTER [4-39](#), [5-8](#)

ATMSAP [4-39](#)

LIF [4-40](#)

Download file [4-17](#), [4-37](#)

Download filename [4-18](#), [4-62](#)

Dump file [4-17](#), [4-37](#), [4-62](#)

Dump filename [4-18](#)

## E

End-of-file mark [4-137](#)

Expand [2-15](#), [4-23](#)

## F

Fabric Status [4-127](#)

Fault tolerance [2-16/2-18](#)

Firmware [4-62](#)

code [4-63](#)

file [4-17](#), [4-37](#), [4-62](#), [4-63](#)

filename [4-18](#)

microcode [4-17](#), [4-62](#)

FTP [2-9](#)

## G

G4SA

naming convention for [3-4](#)

G4SA adapter object

naming convention for [3-6](#)

General receive stats, E4SA [4-91](#)

Generic processes [4-6](#)

## H

Hardware MAC address [4-52](#)

## I

INFO command

ADAPTER [4-41](#)

ATMSAP [4-44](#)

LIF [4-46](#)

MON [4-49](#)

PIF [4-50](#)

SAC [4-60](#), [5-9](#)

Installation [3-1](#)

IOMF CRU

naming convention for [3-2](#), [3-4](#)

IPX/SPX

See NonStop IPX/SPX

## J

JUMBOFRAME [4-34](#)

## L

LAN

configuration [3-9](#)

service providers [2-7](#)

LAN Manager

See LANMAN

LAN manager (LANMAN) process

naming convention for [3-2](#), [3-6](#)

LAN Monitor

See LANMON and MON

LAN monitor (LANMON) process

naming convention for [3-2](#), [3-6](#)

LANMAN [2-14](#)

aborting [5-9](#)

altering [5-9](#)

backup [3-1](#), [5-9](#)

backup process [4-123](#), [4-124](#)

primary [3-1](#), [5-9](#)

primary process [4-123](#), [4-124](#)

process ID [4-123](#), [4-124](#)

LANMON [4-74](#), [4-135](#)  
    adapter ownership [3-1](#)  
    See also MON  
LEC object [2-14](#)  
LENTY object [2-14](#)  
LIF  
    aborting [4-10](#)  
    adding [4-23](#), [4-69](#)  
    altering access list [4-29](#)  
    assigning to PIFs [5-3](#)  
    definition of [2-15](#)  
    deleting [4-40](#)  
    display of names [4-74](#)  
    halting operation of [4-10](#)  
    hierarchy [2-14](#)  
    INFO command [4-46](#)  
    initialization of [3-1](#)  
    MAC address [4-48](#)  
    name of [4-46](#), [4-49](#)  
    naming convention for [3-4](#)  
    naming convention for (H-Series) [3-6](#)  
    PIF association [4-39](#), [4-79](#), [4-110](#),  
    [4-120](#), [4-130](#)  
    relationship to SLSA subsystem [2-10](#)  
    SCF object [2-11](#)  
    starting [4-79](#)  
    TRACE filename [4-111](#), [4-115](#), [4-116](#),  
    [4-117](#), [4-119](#)  
LISTNER process  
    naming convention for (G-Series) [3-4](#)  
LISTOPENS command  
    LIF [4-63](#)  
Local Area Network  
    See LAN  
Logical interfaces  
    See LIF  
Loss of a processor [2-17](#)  
Loss of a ServerNet fabric [2-16](#)  
Loss of access to a SAC [2-16](#), [2-17](#)  
Loss of an adapter [2-18](#)

Loss of media [2-18](#)

## M

MAC address [4-48](#)  
Management tasks [5-1](#)  
Manager process [4-65](#), [4-71](#)  
Managing the SLSA subsystem [5-1](#)  
Microcode  
    application [4-17](#), [4-62](#), [4-63](#), [4-127](#)  
    firmware [4-17](#), [4-62](#)  
MON  
    INFO command [4-49](#)  
    naming [4-71](#)  
    starting [4-80](#)  
    STATUS command [4-111](#)  
    VERSION command [4-149](#)  
MPC object [2-14](#)

## N

NAMES command [5-2](#)  
    ADAPTER [4-67](#)  
    ATMSAP [4-67](#)  
    LIF [4-69](#)  
    MON [4-71](#)  
    null [4-64](#)  
    PIF [4-71](#)  
    PROCESS [4-72](#)  
    SAC [4-75](#)  
Naming conventions [3-2/3-9](#)  
    adapters [3-7](#)  
    ATMSAP [3-8](#)  
    LIFs [3-8](#)  
    PIFs [3-9](#)  
    SACs [3-9](#)  
Network interface [4-52](#), [4-54](#), [4-55](#), [4-56](#),  
    [4-58](#), [4-59](#)  
NonStop IPX/SPX [2-9](#)  
NonStop K-series system [2-15](#)

## O

Object hierarchy [2-14](#)

Objects [2-11/2-15](#)

## P

Parallel Library TCP/IP [2-5](#)

PIF [2-14](#)

    aborting [4-12](#)

    altering [4-32](#)

    definition of [2-15](#)

    display of names [4-74](#)

    effect of ABORT LIF on [4-10](#)

    INFO command [4-50](#)

    initialization of [3-1](#)

    LIF association [4-79](#), [4-110](#), [4-120](#),  
    [4-130](#)

    MAC address [4-52](#)

    names [3-9](#), [4-48](#), [4-50](#)

        assigning [5-2](#)

    SAC subordination [4-39](#)

    SCF object [2-11](#)

    starting [4-82](#)

    stats [4-85](#)

    stopping [4-131](#)

    TRACE command [4-138](#)

PMF CRU

    configuration form [3-10](#)

        example [3-10](#)

    naming convention for [3-2](#), [3-4](#)

Preconfiguration [3-2](#)

PROCESS

    naming [4-72](#)

    STATUS command [4-123](#)

    TRACE command [4-142](#)

    VERSION command [4-151](#)

Processorid [3-2](#), [3-5](#)

Processors [5-1](#)

PROM [4-37](#)

PTRACE [4-135](#)

PVC object [2-14](#)

## Q

QIO subsystem [2-8](#)

QOSSET object [2-14](#)

## R

RESET command

    ADAPTER [4-76](#)

## S

SAC

    aborting [4-14](#)

    access [4-126](#)

    altering access list [4-36](#), [5-8](#)

    definition of [2-15](#)

    display

        access list [4-60](#)

        names [4-74](#)

    halting [4-13](#)

    INFO command [4-60](#)

    information

        configurable [4-60](#)

        nonconfigurable [4-60](#)

    initialization of [3-1](#)

    names [3-9](#)

        assigning [5-2](#)

    naming [4-75](#)

    object subordination [4-133](#)

    ownership [4-37](#), [4-63](#), [4-127](#)

    PIF subordination [4-132](#)

    SCF object [2-11](#)

    ServerNet ID [4-63](#)

    starting [4-82](#)

    STATUS command [4-125](#)

    status information [4-126](#)

        Fabric Status [4-127](#)

    stopping [4-38](#)

SAC (continued)  
TRACE command [4-145](#)  
SAC object [2-14](#)  
SAN [2-8](#)  
SCF  
generic processes [4-6](#)  
nonsensitive commands [4-3](#)  
object hierarchy [4-5](#)  
objects [2-11/2-15](#)  
operational states [4-6](#)  
persistence [2-11](#)  
sensitive commands [4-3](#)  
SCF objects [4-5](#)  
ServerNet addressable controllers  
See SAC  
ServerNet ID [4-61](#), [4-63](#)  
ServerNet WAN concentrator [2-9](#)  
Slot [3-3](#)  
SLSA subsystem  
components of [2-10](#), [4-5](#)  
START command  
ADAPTER [4-17](#), [4-76](#), [5-6](#)  
ATMSAP [4-77](#)  
LIF [4-79](#)  
affect on PIF [4-79](#)  
MON [4-79](#)  
PIF [4-13](#), [4-80](#)  
SAC [4-37](#), [5-2](#)  
Start-up sequence, SLSA [3-1](#)  
STATISTICS command  
ATMSAP [4-83](#)  
PIF [4-84](#)  
STATUS ADAPTER command  
use in procedure [5-6](#)  
STATUS ATMSAP command [4-107](#)  
STATUS command  
ADAPTER [4-106](#), [5-6](#)  
ATMSAP [4-107](#)  
LIF [4-109](#)  
MON [4-111](#)

STATUS command (continued)  
PIF [4-13](#), [4-113](#), [5-6](#)  
PROCESS [4-123](#)  
SAC [4-125](#), [5-6](#)  
using [5-3](#)  
STOP command  
ADAPTER [4-128](#)  
ATMSAP [4-129](#)  
LIF [4-130](#), [5-7](#)  
PIF [4-131](#)  
SAC [4-133](#)  
Subsystem Control Facility  
See SCF  
SWAN concentrator  
naming convention for [3-2](#)  
SWAN concentrator, naming  
convention [3-6](#)  
SWITCH command  
PROCESS [4-134](#)  
system area network  
See SAN  
System-load [3-1](#), [4-37](#)

## T

TCP/IP [2-9](#)  
Parallel Library TCP/IP [2-5](#)  
TCP/IP process  
naming convention for [3-2](#)  
naming convention for (G-Series) [3-4](#)  
naming convention for (H-Series) [3-6](#)  
TELSERV [2-9](#)  
TELSERV process  
naming convention for (G-Series) [3-4](#)  
Trace [2-12](#)  
TRACE command  
MON [4-135](#)  
PIF [4-138](#)  
PROCESS [4-142](#)  
SAC [4-145](#)  
Trace data [4-136](#)

Trace records [4-136](#), [4-139](#), [4-143](#), [4-146](#)

## V

Versatile I/O (VIO) enclosure

adapter type [4-16](#)

ADD ADAPTER [4-15](#), [5-1](#)

configuration file [3-12](#)

overview [2-12](#)

VERSION command

MON [4-149](#)

null [4-150](#)

PROCESS [4-151](#)

## W

WAN

WAN Wizard Pro [1-8](#), [5-1](#)

Wide area network (WAN)

See WAN

## Special Characters

\* [4-41](#)