DEC Network Integration Server

Management

Order Number: AA-PNKMH-TE

Revision/Update Information: This is a revised manual

Software Version: DECNIS™ V4.0

First Printing, March 1992
Revised, October 1996

While Digital believes the information included in this publication is correct as of the date of publication, it is subject to change without notice.

Possession, use, or copying of the software described in this documentation is authorized only pursuant to a valid written license from Digital or an authorized sublicensor.

Digital Equipment Corporation makes no representations that the use of its products in the manner described in this publication will not infringe on existing or future patent rights, nor do the descriptions contained in this publication imply the granting of licenses to make, use, or sell equipment or software in accordance with the description.

The following are trademarks of Digital Equipment Corporation: Bookreader, clearVISN, DDCMP, DEC, DECbrouter, DECmcc, DECnet, DECNIS, DECrouter, DECwindows, Digital, DNA, FLOWmaster, OpenVMS, Packetnet, PATHWORKS, PDP, POLYCENTER, UNIBUS, VAX, VAXcluster, VAXELN, VMS, VT, and the DIGITAL Logo.

AppleTalk is a registered trademark of Apple Computer, Inc.

IBM is a registered trademark of International Business Machines Corporation.

MS–DOS is a registered trademark of Microsoft corporation.

NetBIOS is a trademark of Micro Computer Systems, Inc.

Netview is a registered trademark of International Business Machine Corporation.

NetWare is a registered trademark of Novell, Inc.

OSI is a registered trademark of CA Management, Inc.

TransLAN and Vitalink are registered trademarks of Vitalink Communications Corporation.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company, Ltd.

Ungermann-Bass is a registered trademark of Ungermann-Bass, Inc.

Windows NT is a trademark, and Windows, MS–DOS, and Windows 95 are registered trademarks of Microsoft Corporation.

All other trademarks and registered trademarks are the property of their respective holders.

The Digital implementation of OSPF is an adaptation of the OSPF implementation developed by the University of Maryland, College Park, Maryland. Copyright © 1989, 1990, 1991, 1992, University of Maryland.

Permission to use, copy and modify the software and its documentation is granted provided that this copyright notice and these terms shall appear in all copies of the software and its supporting documentation. The origin of this software may not be misrepresented, either by explicit claim or by omission.

The Software is provided "AS IS" and without any express or implied warranties, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

This document was prepared using VAX DOCUMENT, Version 2.1.

Contents

2 Using a Console Terminal on the DECNIS

3 Configuring Nonvolatile Memory Dynamically

4 Configuring SNMP on the DECNIS

5 Setting Up Event Logging on the DECNIS

6 System-Level Management

7 Packet Prioritization on the DECNIS

8 Connections to the Network

Part II Managing Routing

9 Managing Routing: General

10 IP Routing

11 IP Route Propagation and Filtering

12 BGP Route Propagation and Filtering

13 IP Packet Filtering

14 IP Multicasting

15 IP Standby

16 DECnet/OSI Routing

17 AppleTalk and IPX Routing

Part III Managing X.25

18 Managing X.25 Connections and X.25 Gateway Functions

19 X.25 Relay

20 X.25 Security

Part IV Managing Bridging

21 Setting Up the DECNIS as a Bridge

22 Bridge Filtering on the DECNIS

Part V Appendixes

A Management Modules Used by the DECNIS

B Characteristic Attributes Supported by the DECNIS

C NCL Commands to Set Up Event Sinks

D Common Protocol Values and Formats

E Characteristic Values of the Default X25 ACCESS Template

F Connectivity Tables

G Example NCL Scripts

Index

Figures

Tables

Preface

This manual describes how to manage the software installed on the DEC™ Network Integration Server (referred to throughout this manual as the DECNIS). It describes how to manage routing, X.25 gateway and relay functions, and bridging.

Audience

This manual is intended for network managers.

This manual assumes that you understand and have some experience of:

- Local Area Networks (LANs)
- Wide Area Networks (WANs)
- X.25 (if using the CCITT X.25 protocols)
- OpenVMS™ (if using an OpenVMS load or management host)
- Digital™ UNIX® (if using a Digital UNIX load or management host). (Digital UNIX was formerly called DEC OSF/1.)
- IBM®-compatible Personal Computers running MS–DOS® (if using the MS–DOS PC as a load or management host)
- IBM®-compatible Personal Computers running Windows 95® or Windows NT™ (if using either of these types of system as a load or management host).

Associated Documentation

Product Documentation

- *DEC Network Integration Server Introduction and Glossary*
- *DEC Network Integration Server Installation and Configuration for OpenVMS and Digital UNIX*

- *DEC Network Integration Server Configuration and Management from MS–DOS PCs*
- *clearVISN™ DECNIS Configurator User Guide*
- *DEC Network Integration Server Configuration and Loading*
- *DEC Network Integration Server Problem Solving*

  This is only available on line, as follows:

  - On DECnet/OSI systems, in Bookreader™ format.
  - On Windows 95/NT PCs, as a Windows® help file.
  - On MS–DOS PCs (non-Windows), as a series of text files.
- *DECNIS Event Messages* (supplied on line as a text file)
- *DECNIS Release Notes* (supplied on line as a text file)

Hardware Documentation

The following documents are supplied with the DECNIS hardware:

- *Installation and Service Manual*
- *Configuration Card*

The following documents are supplied with each Network Interface Card:

- *Cabling Instructions and Specifications* card
- *Problem Solving* card
- *Configuration* card

Related Documentation

- NCL online help

  This describes the NCL commands that you can use to manage the DECNIS.
- Network management documentation for the load-host operating system you are using.
- *Common Trace Facility (CTF) Use* manual

  This manual is part of the OpenVMS documentation set, and describes how to use the Common Trace Facility for problem solving.
- *Network Information* (supplied on line)

  This supplies profile information about all the public Packet Switching Data Networks that Digital supports.

- *X.25 Security* manual

  This manual explains the underlying concepts of X.25 security. You can order this manual through your local Digital office.

- *Bridge and Extended LAN Reference* manual

  This manual provides a general description of bridging and extended LANs. You can order this manual through your local Digital office.

- RFCs (for IP routing)

  RFCs are the working notes for the internet research and development community. These notes are available in a three-volume set, the *DDN Protocol Handbook*, which can be ordered from the following address:

  Network Solutions, Inc.
  Attn: InterNIC Registration Service
  505 Huntmar Park Drive
  Herndon, VA 22070, USA
  Tel. 1-800-444-4345 or 619-455-4600

Returning Comments About this Documentation

We would like to know what you think about the DECNIS documentation set and online help.

If you have any comments, or suggestions, please return them in any of the following ways:

- Send an electronic mail message to the Internet address books@reo.mts.dec.com
- Send an electronic mail message to the X.400 address S=IDC BOOKS; O=digital; OU1=reo; P=digital; A=CWMail; C=gb
- Send a fax to (+44)134 206018

Conventions

The following conventions are used in this manual:

Italics	This indicates variable information.
decnis	This indicates that you should substitute the node name of the DECNIS. If you are using DECdns or the local namespace, enter the registered name.
DECNIS	DEC Network Integration Server

PC An IBM-compatible personal computer

Prompts The following prompts precede commands that you enter:

For OpenVMS: `$`
For Digital UNIX: `#`
For MS–DOS PCs: `C:\`
For the DECNIS console: `console>`
For NCL: `NCL>`

Part I

Managing the System

This part contains information on managing those aspects of the DECNIS that are not protocol specific.

It contains the following chapters:

- Chapter 1 describes the DECNIS management model and the tools you can use to manage the DECNIS.
- Chapter 2 describes how to use the DECNIS console.
- Chapter 3 describes how to modify the contents of DECNIS nonvolatile (flash) memory on the MPC-II and MPC-III.
- Chapter 4 describes how to set up SNMP to manage the DECNIS.
- Chapter 5 explains how to set up event logging from the DECNIS.
- Chapter 6 describes system-level management.
- Chapter 7 describes how to set up the packet prioritization function of the DECNIS.
- Chapter 8 describes how to set up synchronous and LAN connections.

1

Tools for Managing the DECNIS

1.1 Introduction: SNMP and DEC CMIP

1.1.1 Two Management Protocols

The DECNIS supports two management protocols:

- SNMP (Simple Network Management Protocol)
- DEC CMIP (Common Management Information Protocol)

1.1.2 SNMP (Simple Network Management Protocol)

SNMP is an Internet standard protocol, specified in RFC 1157.

You can use a network management station running SNMP to monitor and change a range of the functions of the DECNIS.

You can only use SNMP to manage a running DECNIS: any changes you make to its configuration will be lost if the DECNIS is rebooted.

Chapter 4 describes how to configure the DECNIS to be manageable using SNMP.

1.1.3 DEC CMIP (Common Management Information Protocol)

DEC CMIP is a Digital-proprietary management protocol. It is used to monitor and change all DECNIS functions, either dynamically or statically (see Section 1.3.1.1 and Section 1.3.1.2).

Section 1.2 to Section 1.12 describe the CMIP management model, and the tools that use CMIP.

1.2 Management Modules

All the management information for the DECNIS, including configuration details, is held in a series of modules, each representing a functional part of the system.

1.2.1 Entities

A module usually contains one or more entities, each dealing with a part of that module’s function.

1.2.2 Entity Attributes

Each entity has one or more attributes, the values of which determine the behavior of the relevant DECNIS function.

1.2.3 Example

The Routing module represents the routing functions of the DECNIS. It contains the entities shown in Figure 1–1.

Each ROUTING CIRCUIT entity on the DECNIS has attributes that determine how the corresponding routing circuit behaves.

For example, the value of the HELLO TIMER attribute of a ROUTING CIRCUIT entity called CIRCUIT_1 determines how often the DECNIS will generate router Hello messages on the routing circuit CIRCUIT_1.

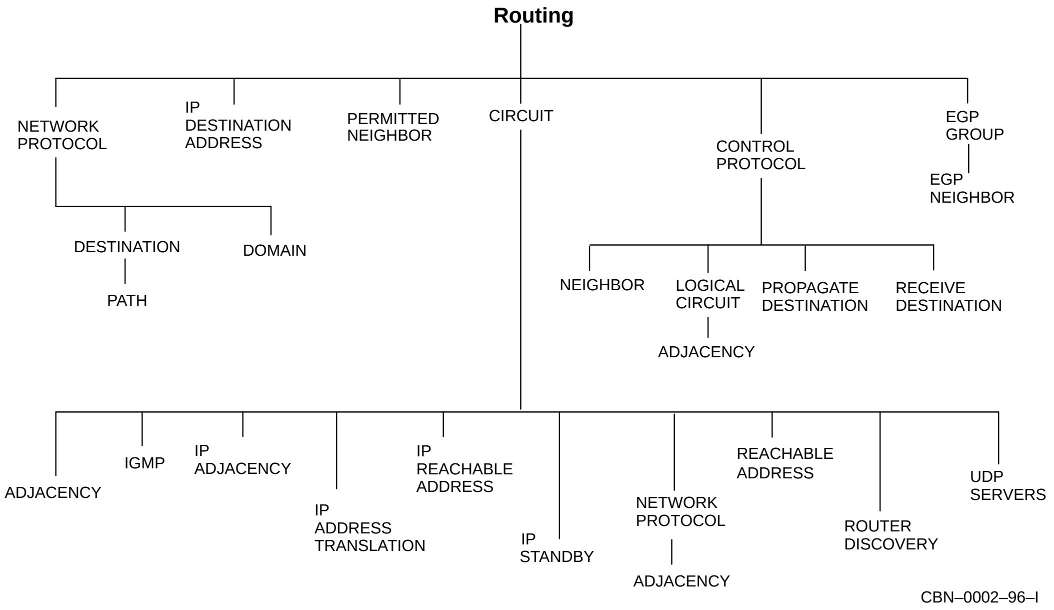
1.2.4 Entities and Management

To manage the DECNIS, you use the management tools described in the remainder of this chapter. These tools manipulate the management modules and entities, and the configuration information they contain.

Use the management tools described to:

- Set up new modules and/or entities.
- Delete existing modules and/or entities.
- Change the characteristics of existing entities.

Figure 1–1 Routing Module



1.2.5 Entities Implemented in the DECNIS

Figure 1–2 shows the management modules implemented in the DECNIS.

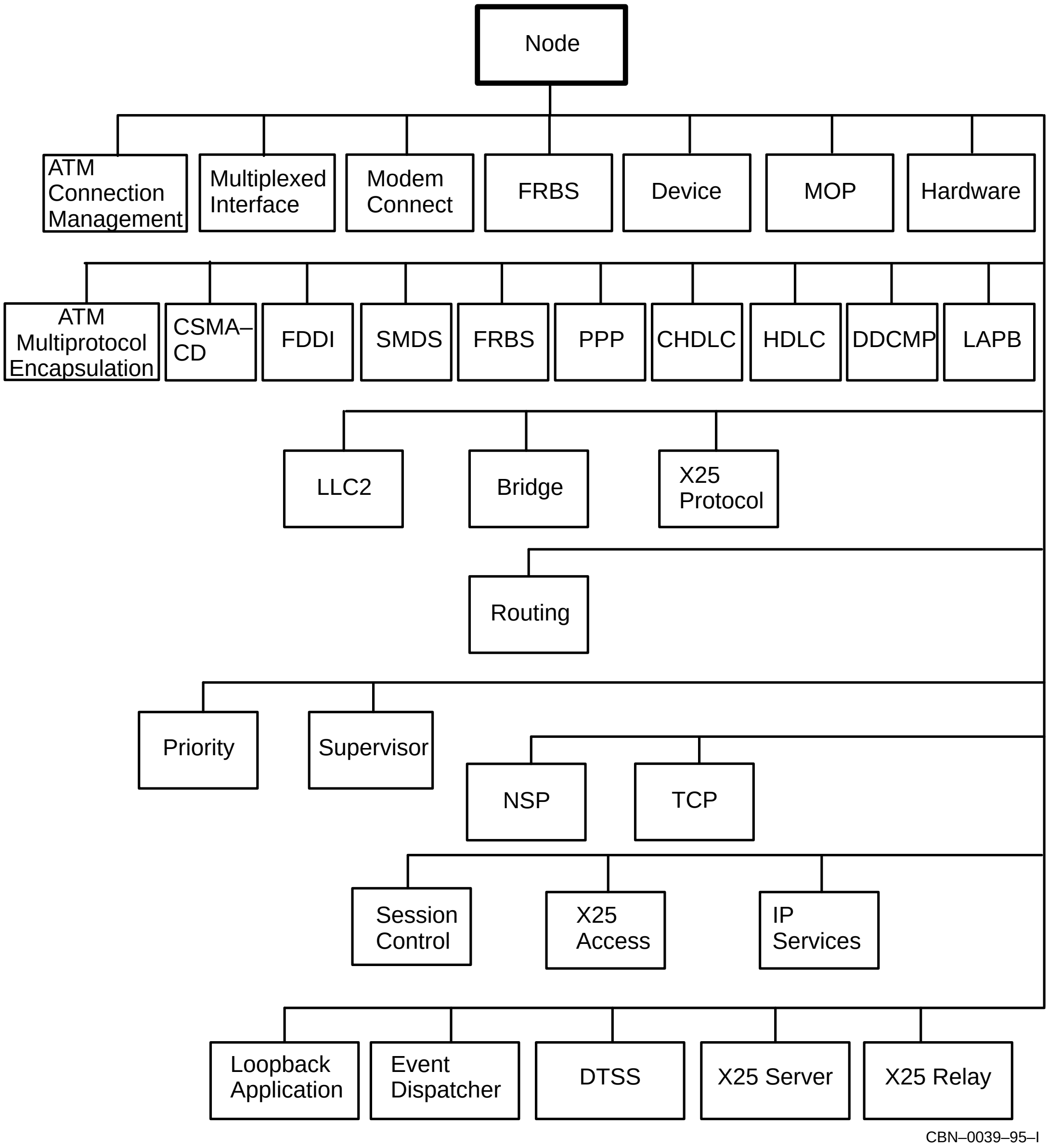
Appendix A describes all the modules and associated entities used by the DECNIS.

Appendix B lists the entities' characteristics, as implemented in the DECNIS.

Figure 9–1, Figure 9–2, Figure 9–3, and Figure 9–4 show the entities associated with a routing circuit, and how they are related.

Figure 21–2, Figure 21–3, and Figure 21–4 show the entities associated with a bridge port, and how they are related.

Figure 1–2 Management Modules Implemented in the DECNIS



1.3 Using NCL

1.3.1 What Is NCL?

The Network Control Language (NCL) is a utility that enables you to configure, manage and monitor your DECNIS on the network. You issue NCL commands to manipulate the DECNIS modules described in Appendix A.

The network management documentation for OpenVMS and Digital UNIX provides information on most of the management modules described in Appendix A. The NCL online help provides more detailed information on some of these modules, and information on the modules not covered there.

Note that on Windows 95/NT load hosts, NCL help is supplied in a Windows help file; refer to the manual *clearVISN DECNIS Configurator User Guide* for more information.

You can issue NCL commands either dynamically or statically, as described in Section 1.3.1.1 and Section 1.3.1.2.

1.3.1.1 Dynamic Management: Definition

Dynamic management means making changes to the configuration of the DECNIS while it is running. To do dynamic management, you issue NCL commands to a running DECNIS. The DECNIS responds immediately to the NCL commands you issue.

When the DECNIS is rebooted, the configuration changes made dynamically are lost.

1.3.1.2 Static Management: Definition

Static management means changing the permanent configuration of the DECNIS. These changes do not take effect until the DECNIS is rebooted.

The permanent configuration consists of the management information that is loaded to the DECNIS. This is the binary version of the **NCL script files**.

The NCL script files are text files containing NCL commands. They are stored on the load host.

Refer to Section 1.5.2 and Section 1.5.3 for more information about the permanent configuration.

1.4 Using Dynamic Management

This section describes the types of system from which you can do dynamic NCL management, and gives general information on issuing NCL commands.

1.4.1 Management Systems

You can issue NCL commands from the following types of system:

- A supported management host. Refer to Section 1.4.1.1.
- The DECNIS console. Refer to Section 2.13.1.

1.4.1.1 Supported Management Hosts

Management hosts can be any of the following:

- OpenVMS or Digital UNIX systems running DECnet/OSI.
- Digital UNIX systems using Transmission Control Protocol (TCP) as the transport protocol. Refer to Section 1.9.
- IBM-compatible Personal Computers (PCs) running MS–DOS.
- IBM-compatible PCs running Windows 95 or Windows NT.

1.4.1.2 General Requirements for Management Hosts

Supported management hosts must meet the following requirements:

- The DECNIS software must be installed on the host.
- The DECNIS must be reachable from the host over the network.

A management host is typically a load host, but is not required to be.

1.4.2 Starting NCL and Issuing Commands

To issue NCL commands to manage the DECNIS, follow these steps:

1. Log on to a suitable management host or the DECNIS console.
2. Start NCL on the system, as follows:

To start NCL on...	Refer to...
OpenVMS and Digital UNIX systems	Network management documentation for the host
MS–DOS PCs	*DECNIS Configuration and Management from PCs*
Windows 95 or Windows NT PCs	*clearVISN DECNIS Configurator User Guide*
The DECNIS console	Section 2.13 in this manual

3. Issue the required NCL commands.

1.4.3 Specifying the DECNIS in NCL Commands

When entering NCL commands, you need to identify the DECNIS node to which the command will apply. You can include the node name in the command, or set up the DECNIS as a default node, as described in Section 1.4.4.

The node name itself can be specified in different ways, depending on the type of management host from which you are issuing commands, and the type of network being used to transport the commands to the DECNIS. Table 1–1 shows the different types of node specification. The **Example Syntax** column gives example commands; however, note that it does not show all possible variations in syntax.

Table 1–1 Node Specifications in NCL commands

Management Host	Node Specifications	Example Syntax
DECnet/OSI host	A DECdns or Local Name Service name	`SHOW NODE ORG:.SOUTH.SALES`
MS–DOS PC	DECnet Phase IV address	`SHOW NODE 1.100`
	DECnet Phase IV node name	`SHOW NODE nis100`
Windows 95/NT PC Digital UNIX host	IP address	`SHOW NODE 16.36.16.100`
	IP Domain Name Service node name	`SHOW NODE nis100`

1.4.4 Using a Default Node for NCL Commands

If you wish to issue several NCL commands to the same DECNIS from a management host, set default to the remote system. This section describes how to do this on supported load hosts.

1.4.4.1 MS–DOS PC, Digital UNIX or OpenVMS Hosts

- On MS–DOS PC or Digital UNIX hosts, enter the following command:

  ```
  NCL> SET NCL DEFAULT ENTITY NODE decnis/username/password
  ```

- On an OpenVMS system, enter the following command:

  ```
  NCL> SET NCL DEFAULT ENTITY NODE decnis"username password"
  ```

where *decnis* is the node name of the DECNIS, and *username* and *password* are the network management user name and password. Section 6.2.2 describes this user name and password.

1.4.4.2 Windows 95 or Windows NT Hosts

On Windows 95/NT hosts, follow these steps:

1. On the toolbar, go to the **Tools** menu and select **Start NCL**.
2. On the NCL window, click the **Defaults** button.
3. Check the box next to the Node Name or Address field, and then enter the node name or address.
4. If you have previously set up a network management user name and password, check the boxes next to the User Name and Password fields. Then, enter the same user name and password that you entered for the network management user name and password.

   Note that you set up a network management user name and password on the Security tab page under the **System** button.
5. Click **OK**.

1.5 Using Static Management

This section describes the tasks needed to carry out static management on the DECNIS. These tasks are:

1. Create an NCL script for the DECNIS.
2. Create a CMIP file.
3. If you wish, create a combined file.
4. Load the CMIP file and image, or the combined file, to the DECNIS.

1.5.1 Systems Used for Configuration and Loading

The system used for DECNIS configuration and loading is referred to as a load host. A load host can be any of the following types of system:

- OpenVMS or Digital UNIX systems running DECnet/OSI.
- IBM-compatible PCs running MS–DOS.
- IBM-compatible PCs running Windows 95 or Windows NT.

1.5.2 NCL Scripts

An NCL script is an ASCII file containing NCL commands. The NCL script file resides on the load host for your DECNIS.

1.5.2.1 Creating NCL Scripts

There are two ways to create NCL scripts for the DECNIS:

- Using the DECNIS text-based configurator or the clearVISN™ DECNIS configurator. Each of these utilities generates a master NCL script from information you enter about your DECNIS configuration. This is the recommended method. Refer to Section 1.6 for more information about the configurators.
- Entering commands directly in an NCL script file. This is only recommended for the user NCL script files (also called extra files). The purpose of the user NCL script files is to allow you to configure features that are not supported in the configurators. Refer to Section 1.5.6 for details.

1.5.3 CMIP Files and Combined Files

You must convert the NCL script(s) to a **CMIP file** before you can load it to the DECNIS. You can, if you wish, combine the CMIP file, the system image and the profile files into a single file, called the **combined file**.

CMIP File: Definition

A CMIP file is a binary version of the NCL script(s). It is sometimes referred to as a configuration file. It can be loaded separately, or as part of a combined file. Section 1.5.4 describes how to create a CMIP file.

Combined File: Definition

A combined file is a single file that contains the system image, CMIP file and profile files. It is sometimes referred to as the image/CMIP/profile file. Section 1.5.5 describes how to create a combined file.

If you create a combined file, it is always loaded into flash memory.

Note that you can modify the combined file before you load it; refer to Section 1.8.

System Image: Definition

The system image is the software for the DECNIS. It is always loaded into flash memory, whether it is loaded separately or as part of the combined file.

The system image has the following name:

- On OpenVMS or Digital UNIX load hosts:

  NIS040.SYS
- On MS–DOS or Windows 95/NT load hosts:

  install-directory\COMMON\NISV40\SYSTEM

The DECNIS system image is a double image; it contains two internal images:

- One that only supports MPC–I features.
- One that supports MPC–II/III features.

The DECNIS only loads one of the internal images into nonvolatile memory. Which internal image is loaded depends on which management processor card is installed.

For more information, refer to the DECNIS installation manual for your load host.

1.5.3.1 Dynamically Updating Flash Memory

Once you have loaded the DECNIS, you can use DECNIS console commands to add one or more CMIP files to flash memory. You can do this regardless of whether you have loaded a combined file or a separate CMIP file and system image. Refer to Chapter 3 for details.

1.5.4 Creating a CMIP File

This section describes how to convert an NCL script to a CMIP file.

1.5.4.1 MS–DOS PC, OpenVMS or Digital UNIX Hosts

On these hosts, you can convert the NCL script files to a CMIP file within the DECNIS text-based configurator. Alternatively, use the following commands:

- MS–DOS PCs:

  ```
  C:\install-directory\NIS\NIS_SCPL client-name
  ```

- OpenVMS hosts:

  ```
  $ @SYS$MANAGER:NIS$SCRIPT_COMPILE NCL-script
  ```

- Digital UNIX hosts:

  ```
  # /usr/lib/dnet/nis_script_compile NCL-script
  ```

where *NCL-script* is the name of the NCL script.

1.5.4.2 Windows 95 or Windows NT Hosts

On Windows 95/NT hosts, you can convert the NCL script file to a CMIP file within the clearVISN DECNIS configurator. Refer to Section 1.6.4.

1.5.5 Creating a Combined File

This section describes how to create a combined file.

1.5.5.1 MS–DOS PC, OpenVMS or Digital UNIX Hosts

On these hosts, you can create a combined file within the DECNIS text-based configurator. Alternatively, create a combined file by following these steps:

1. Create a CMIP file, as described in Section 1.5.4.
2. To combine the CMIP file and the image and profile files, enter the following command:
   - MS–DOS PCs:

     ```
     C:\install-directory\NIS\NIS_ICMP NISV40 client-name
     ```

   - OpenVMS load hosts:

     ```
     $ @SYS$MANAGER:NIS$IMAGE_COMPRESS.COM NIS040 client-name
     ```

   - Digital UNIX load hosts:

     ```
     # /usr/lib/dnet/nis_combine nis040 client-name
     ```

   where *client-name* is the load client name of the DECNIS and *install-directory* is the installation directory.

1.5.5.2 Windows 95 or Windows NT Hosts

On Windows 95/NT hosts, you create the combined file within the clearVISN DECNIS configurator. Refer to Section 1.6.4.

1.5.6 User NCL Script Files

User NCL files are extra NCL script files that you use to enter NCL commands in addition to those generated by the configurators.

Empty user NCL script files are generated by the DECNIS configurators. The configurator compiles the user NCL script files and the master NCL script to create a CMIP file.

Note

You are advised not to change the permanent configuration of the DECNIS by editing the master NCL script produced by the DECNIS configurator. Any changes you make to the master NCL script will be lost when you next run the configurator.

Note that it is not possible to edit the master NCL script produced by the clearVISN DECNIS configurator.

1.5.6.1 Types of User NCL Script File

The user NCL script files are described below. See *DECNIS Configuration and Loading* for the file specifications on each type of load host.

User NCL Script Files	Purpose
CREATE user NCL script	Insert NCL CREATE commands to create new entities.
SET user NCL script	Insert NCL commands to change the values of an entity's characteristic attributes, such as SET, ADD, REMOVE, and BLOCK.
ENABLE user NCL script	Add NCL ENABLE commands, to enable the entities created by commands in the CREATE user NCL script.

1.5.6.2 Editing User NCL Script Files in the clearVISN DECNIS Configurator

You can edit the user NCL script files within the clearVISN DECNIS configurator, as described in the manual *clearVISN DECNIS Configurator User Guide*.

You cannot edit the user NCL script files within the DECNIS text-based configurator.

1.5.6.3 How the DECNIS Uses the User NCL Script Files

The user NCL script files are invoked when you convert the master NCL script to a CMIP file, as shown in Figure 1–3.

1.5.6.4 Example

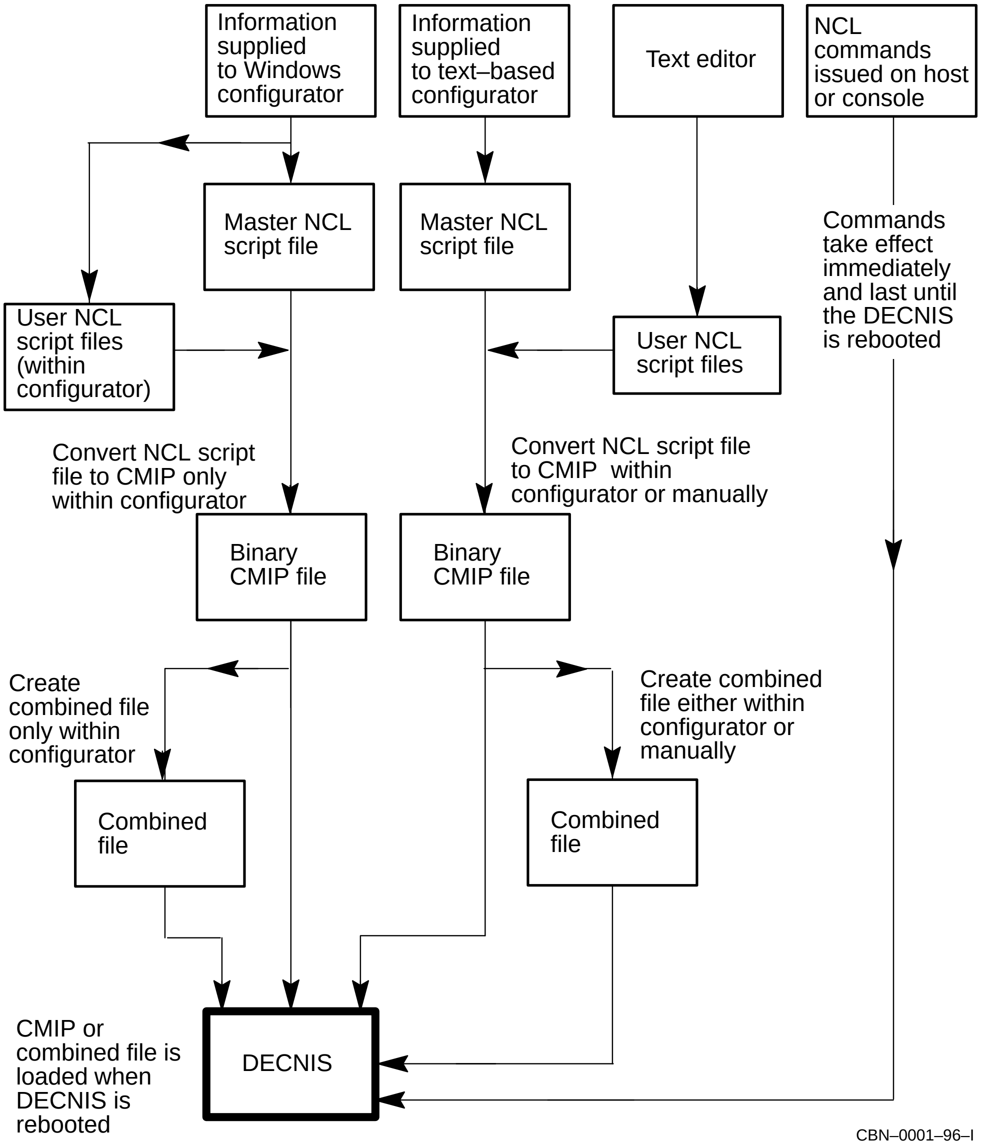
To change the value for the acknowledge timer on an HDLC data link, follow these steps:

1. Find the name of the HDLC link, by examining the master NCL script.
2. Find the command required to change the value of the acknowledge timer:

   ```
   NCL> SET HDLC LINK link-name ACKNOWLEDGE TIMER n
   ```

3. Insert this command in the SET user NCL script.
4. Create a configuration load file; refer to Section 1.5.3 for details.
5. Reboot the DECNIS.

Figure 1–3 Managing the DECNIS Using NCL and the DECNIS Configurators


Information supplied to Windows configurator
Information supplied to text–based configurator
Text editor
NCL commands issued on host or console
Master NCL script file
Master NCL script file
Commands take effect immediately and last until the DECNIS is rebooted
User NCL script files (within configurator)
User NCL script files
Convert NCL script file to CMIP only within configurator
Convert NCL script file to CMIP within configurator or manually
Binary CMIP file
Binary CMIP file
Create combined file only within configurator
Create combined file either within configurator or manually
Combined file
Combined file
CMIP or combined file is loaded when DECNIS is rebooted
DECNIS
CBN–0001–96–I

1.6 Using the Configurators

1.6.1 Introduction

The following configuration utilities are provided to help you configure the DECNIS software:

Configuration Utility	Description	Supported Hosts
Load-host configurator	A text-based, menu-driven program, running on a VT220 terminal, used to provide information needed for loading. It is used in conjunction with the DECNIS text-based configurator.	DECnet/OSI for OpenVMS Alpha DECnet/OSI for OpenVMS VAX Digital UNIX MS–DOS PCs
DECNIS text-based configurator	A text-based menu-driven program, running on a VT220 terminal, used to provide information about the DECNIS configuration. It is used in conjunction with the load-host configurator.	DECNET/OSI for OpenVMS Alpha DECNET/OSI for OpenVMS VAX Digital UNIX MS–DOS PCs
clearVISN DECNIS configurator	A Windows graphical user interface (GUI) program that combines the functions of the load-host configurator and the DECNIS text-based configurator.	Windows 95/NT PCs

Note that the DECNIS text-based configurator and the clearVISN DECNIS configurator do not support identical functions and facilities. Refer to the manuals *DECNIS Configuration and Loading* and *clearVISN DECNIS Configurator User Guide* for details.

1.6.2 DECNIS Configurators Description

The DECNIS text-based configurator and the clearVISN DECNIS configurator enable you to supply information to define your DECNIS configuration. Based on this information, the configurators can create:

- A master NCL script.
- A CMIP file or combined file which can be loaded to the DECNIS.

1.6.3 Modifying a Configuration: DECNIS Text-Based Configurator

Follow these steps to modify a DECNIS configuration within the DECNIS text-based configurator:

1. Specify Modify from the Main Menu of the DECNIS text-based configurator, and select the name of the DECNIS to be modified.

   Result: The DECNIS will read in the existing configuration information for this DECNIS from a data file. You cannot modify this data file manually.

2. Select a section and enter additional information (for example, add a routing circuit), or change existing information (for example, change the packet size for an X.25 DTE).

3. When you have finished modifying the configuration, select Continue to New Section from any Options menu.

   Result: You will see the NCL Script screen.

4. Select Create an NCL Script.

   Result: The next screen provides the following choices:

   - Create a CMIP file or a combined file from the NCL script

     Select this if you do not want to edit the user NCL script files (see Section 1.5.6). When the DECNIS text-based configurator has created either a separate CMIP file or a combined file, you will return to the Main Menu.

   - Return to Sections Menu

     Select this if you want to check information you provided previously, or change some of your answers and create another NCL script.

   - Return to Main Menu

     Select this if you want to create or modify a configuration for another DECNIS.

   - Exit From the configurator

     Select this if you want to edit the user NCL script files. After you have edited the user NCL script files, you can enter commands to compile the master NCL script into a CMIP file and (if desired) create a combined file. Refer to Section 1.5.4 or Section 1.5.5.

1.6.4 Modifying a Configuration: clearVISN DECNIS Configurator

Follow these steps to modify a DECNIS configuration within the clearVISN DECNIS configurator:

1. Select an existing DECNIS configuration from the File menu.
2. On the Main Navigation window, click the button that takes you to the tab pages you want to change.
3. Make your changes on the tab pages, and click OK.
4. Repeat steps 2 and 3 until you have completed your changes.
5. If you want to edit the user NCL script files (see Section 1.5.6), click the **NCL scripts** button on the Main Navigation window. You can then add NCL commands on the Extra tab pages. Note that you cannot edit the NCL commands on the Generated NCL tab page.
6. On the Main Navigation window, click the **System** button.
7. On the Load Options tab page, do the following:
   - Select a Load method under Image Loads: either **From Flash** or **From Network**.

     If you want the configurator to create a combined image/CMIP/profile file, check the **CMIP Script** box under Flash Contents.
   - Click **OK**.
   - On the Main Navigation Window, click the **Compile** button.

1.7 Security for NCL Commands

The DECNIS text-based configurator requires you to set up access control information for the use of NCL, by creating a network management username and password. Once you have created this username and password, you must supply them in any NCL commands you issue for the DECNIS from a host, except for SHOW commands. In addition, if you have not set up a console password, you must supply the network management password when you use Telnet to connect to the DECNIS console.

The clearVISN DECNIS configurator allows you to set up access control information for the use of NCL, but it does not require you to do so.

Refer to Section 6.2 for more information about setting up NCL security. Refer to Section 2.13 for information about setting up access control for console NCL.

1.7.1 Example: Supplying Access Control Information in NCL Commands

This example shows the format used to enter access control information in NCL commands from a host.

You want to create a CSMA/CD circuit called SALT on the DECNIS node ORG:.SOUTH.SALES, which has a username SMITH and password SECRET. This node has the DECnet Phase IV address 56.45. Enter the following command:

- OpenVMS hosts:

```
NCL> CREATE NODE org:.south.sales"smith secret" ROUTING CIRCUIT -
_NCL> salt TYPE CSMA-CD
```

- MS–DOS PCs:

```
ncl> create node 56.45/smith/secret routing circuit -
_ncl> salt type csma-cd
```

- Windows 95/NT PCs:

```
ncl> create node 26.8.8.8/smith/secret routing circuit -
_ncl> salt type csma-cd
```

- Digital UNIX hosts:

```
ncl> create node org:.south.sales/smith/secret routing circuit -
_ncl> salt type csma-cd
```

1.7.2 Setting Default Security for NCL Commands

If you have set up a DECNIS as your default node, as described in Section 1.4.4, you can set default security for NCL on that DECNIS. Once you have set default to the DECNIS, issue the following command:

```
NCL> SET NCL DEFAULT ACCESS BY USER username, PASSWORD password
```

1.8 Modifying the Combined File

MOD_FLSH is a utility that enables you to do the following on a load host:

- Add files to the combined file before it is loaded.
- Delete files from the combined file (or the double system image) before it is loaded.
- Display the contents of the combined file or system image.

For more information about the combined file, refer to Section 1.5.3. The section System Image: Definition in this chapter briefly describes the double system image; for more detailed information, refer to the DECNIS installation and configuration manuals for your load host.

1.8.1 Before You Start

Before you run MOD_FLSH, create a combined file, as described in Section 1.5.5.

1.8.2 Starting MOD_FLSH on MS–DOS PC, OpenVMS or Digital UNIX Hosts

To start MOD_FLSH, enter the following command(s) at the prompt:

- MS–DOS PCs:

  ```
  C:\install-directory\MOD_FLSH combined-file
  ```

- OpenVMS load hosts:

  ```
  $ MOD_FLSH :== $ SYS$SYSTEM:MOD_FLSH.EXE
  $ MOD_FLSH combined-file
  ```

- Digital UNIX load hosts:

  ```
  # /usr/lib/dnet/mod_flsh combined-file
  ```

where *combined-file* is the file specification of the combined file.

1.8.3 Starting MOD_FLSH on Windows 95/NT Hosts

On a Windows 95/NT PC, you can start MOD_FLSH either from the clearVISN DECNIS configurator or from an MS–DOS window.

Starting from the Configurator

The clearVISN DECNIS configurator will only start MOD_FLSH immediately after compiling an NCL script file. To start MOD_FLSH, follow these steps:

1. When you have finished configuring the DECNIS, click the **Compile** button on the Main Navigation window. The configurator will display a status window.
2. When the configurator has finished compiling the NCL script, click the **Modify Image** button on the status window. The configurator will start MOD_FLSH.

Starting from an MS–DOS Window

To start MOD_FLSH outside the configurator, follow these steps:

1. Open an MS–DOS window.
2. At the prompt, enter the command for MS–DOS PCs in Section 1.8.2.

1.8.4 MOD_FLSH Display

When you start MOD_FLSH, it displays information about the combined file or system image, including a list of its files. You can then enter commands at the prompt. For example:

```
   Flash Directory of file "nis1.sys"
Image Major/Minor ID 02.05
Image Name "CO_GE_SYS_ROU"
Identification "DECNIS V4.0  "
Link Date/Time  13 October 96 09:20
Flash Pages 8374, Loaded Pages 8399
V2.5/V3.0 format with two system images
Built by flash_proc 4.0

Contents of Flash Structure:

   Index     Size       Date   Time     Name
     1     3335112   29 Aug 96 15:16   nis_v4.0B
     2     1944520   29 Aug 96 13:11   nis_v4.0
     3       14460   19 Sep 96 15:24   script
     3       12210   30 Sep 96 15:24   script.test
     4        1028   19 Aug 96 14:10   mcnm_prf
     5       27242   19 Aug 96 14:10   X25l2_prf
     6       76072   19 Aug 96 14:10   X25l3_prf
Command (h for help) >
```

1.8.5 Exiting from MOD_FLSH

To exit from MOD_FLSH, enter `Quit` at the command prompt.

1.8.6 Getting Help

To display the list of MOD_FLSH commands, press h and then press Return at the prompt.

1.8.7 Adding a File

To add a file to the combined file, enter the following command:

```
add file-spec flash-name
```

where: *file-spec* is the specification of the file to be added

flash-name is an optional internal name for the file being added

It is strongly recommended that you supply the *flash-name*. This helps ensure that the combined file does not contain several files with the same name.

1.8.8 Deleting a File

To delete a file or image, enter the following command at the prompt:

```
delete index-number
```

where *index-number* is the index number of the file to be deleted. To view the index numbers, enter `list` at the prompt.

Note that you are not allowed to delete the file with the index 1. This file is the default system image.

1.8.9 Extracting a File

MOD_FLSH allows you to copy (extract) one of the files in the combined file to a disk on the load host. To do this, enter the following command:

```
extract index-number file-spec
```

where *index-number* is the index number of the file to be copied and *file-spec* is the file specification that you want to give to the extracted file.

Note the following:

- On OpenVMS load hosts, a file that has been extracted cannot be reloaded separately. For example, if you extract a script file from the combined file, it cannot be loaded from the load host as a separate CMIP file. It can only be loaded as part of a combined file.
- On PC and Digital UNIX load hosts, this does not apply. An extracted file can be loaded as a separate CMIP file.

1.8.10 Displaying the Combined File

To redisplay the contents of the combined file or image, enter either of the following commands:

- `List`
- `Relist`

1.9 Using CMIP Over TCP

You can configure the DECNIS to use TCP as well as NSP as its transport protocol for network management. This enables you to issue NCL commands to manage the DECNIS in an IP-only environment.

NSP is used as the DECNIS transport protocol in DECnet environments.

1.9.1 What TCP Enables You to Do

Setting up TCP on the DECNIS enables you to do the following:

- Issue NCL commands to the DECNIS from a Digital UNIX V3.0 host that is using TCP as its transport. The Digital UNIX host does not need to be a DECnet/OSI system.
- Issue NCL commands from the DECNIS console to manage the following types of system:
  - Digital UNIX V3.0 or later systems.
  - SNA Gateway V2.0 or later systems.
  - Other DECNIS V3.1 or later systems.

CMIP over TCP Not Usable From a DECnet/OSI for OpenVMS Host

You cannot manage the DECNIS over TCP from a DECnet/OSI for OpenVMS host. This is because DECnet/OSI for OpenVMS implements TCP according to RFC 1006, while the DECNIS implementation does not.

1.9.2 Managing the DECNIS Over TCP

Table 1–2 lists the tasks you need to carry out before you can manage the DECNIS from a Digital UNIX system using TCP.

Table 1–2 Setting Up TCP

On the DECNIS, set up TCP.	The DECNIS configurator (text-based or Windows) automatically adds the NCL commands to set up TCP to the master NCL script. Alternatively, you can issue the following NCL commands:
	`CREATE TCP`
	`ENABLE TCP`

On the Digital UNIX system, issue the following NCL command:	`SET NCL TRANSPORT=TCP`

Cannot Disable TCP

Note that there is no NCL command to disable TCP. In order to remove TCP from the DECNIS, you must do the following:

- Remove the CREATE TCP and ENABLE TCP commands from the master NCL script.
- Reload the DECNIS.

However, note that you cannot do this on Windows 95/NT load hosts, as you cannot edit the master NCL script file within the clearVISN DECNIS configurator.

1.9.3 Managing Other Systems from the DECNIS

To set up the DECNIS to manage Digital UNIX and SNA Gateway systems using TCP, follow these steps:

1. Set up TCP on the DECNIS by running the DECNIS configurator. Alternatively, issue the following NCL commands:

   ```
   NCL> CREATE TCP
   NCL> ENABLE TCP
   ```

   Result: You can issue NCL commands over TCP, using an IP address to identify the node being managed.

   If you want to use IP node names as well as IP addresses to identify nodes being managed, continue to the next step.

2. Set up the DECNIS to use the IP Domain Name Server by entering the following NCL commands

   ```
   NCL> CREATE IP SERVICES RESOLVER
   NCL> CREATE IP SERVICES RESOLVER SERVER server-name IP ADDRESS ip-address
   NCL> ENABLE IP SERVICES RESOLVER
   ```

   where *server-name* is the name of an IP Domain Name Server willing to answer queries, and *ip-address* is its IP address.

3. Set TCP as the NCL transport to be used on the DECNIS:

   ```
   NCL>  SET NCL TRANSPORT TCP
   ```

1.9.3.1 Effect of Setting NCL Transport to TCP

Issuing the NCL command SET NCL TRANSPORT TCP affects the way node specifications are interpreted in NCL commands, as follows:

- If a node name is used to specify the node being managed, it is interpreted as an IP node name. If the name cannot be resolved by the IP Domain Name Server, the command will fail.
- If an IP address or DECnet Phase IV address is used to specify the node being managed, it is interpreted either as an IP address or as a DECnet address on the basis of its syntax.

1.9.3.2 Switching Back to DECnet Transport

To switch back to using DECnet transport, issue the following command:

```
NCL>  SET NCL TRANSPORT DECNET
```

1.10 Using the DECnet/OSI Naming Services

1.10.1 Introduction

On a DECnet/OSI network, node name and addressing information can be stored within the DECdns namespace or a local namespaces.

1.10.2 DECdns

DECdns servers in a network store addressing information about all the nodes in the network. DECdns clerk software running on a host system can access this addressing information. In this way, each host system does not have to maintain a database of node addressing information.

1.10.3 Local Namespace

With local namespaces, each node in a network maintains a discrete, local namespace, containing addressing information about the other nodes in the network. You can use local namespaces as well as, or instead of, using DECdns servers in a network.

1.10.4 The DECNIS and the Naming Services

The DECNIS does not use the naming services directly, as it does not contain DECdns clerk software.

However, the load-host and DECNIS text-based configurators can use the naming services to do the following:

- Build the Known Towers database on the DECNIS. This database contains specifications of all the nodes to which the DECNIS sends messages; for example, event sinks. Refer to Section 1.11 for details.
- Register the DECNIS node in the local or DECdns namespace. Refer to the manual *DECNIS Configuration and Loading* for details.

Note that the clearVISN DECNIS configurator does not make use of the naming services.

1.11 Translation of Node Names on the DECNIS

1.11.1 Introduction

Several management tasks, if performed using NCL commands, require you to enter the names of other nodes in the network (for example, in Section 5.3 you enter the node name of the event sink).

For each of these node names, you must set up information to allow the DECNIS to translate the name to a complete specification of how the remote node can be reached. This is necessary even if the network is using DECdns and/or the local namespace to translate node names into addressing information.

The complete node specification is known as a **tower set**.

1.11.2 Tower Sets

A tower set is a list of the protocols and addresses by which the DECNIS can reach another node in the network.

You allow the DECNIS to reach another node in the network by entering the tower set of the remote node in the Known Towers database of the DECNIS.

1.12 Entering Tower Sets in the Known Towers Database

You can use either DECNIS configurator to create tower sets automatically. You can also issue NCL commands to create tower sets if you wish.

1.12.1 Entering Tower Sets Using the DECNIS Text-Based Configurator

The DECNIS text-based configurator can create tower sets in two ways:

- Using node specifications in the DECdns or local namespace.
- Using DECnet Phase IV or NSAP addresses that you supply.

1.12.1.1 Using a Naming Service

The load-host configurator and the DECNIS text-based configurator can use DECdns or the local namespace to find node specifications. In order to do this, the configurators must be running on a host which meets the following requirements:

- It must be running DECdns clerk software.
- **One** of the following must be true:
  - A suitable DECdns server is available.
  - A local namespace is set up on the host.

Procedure

In the load-host configurator, you are asked whether or not you want to use a naming service to generate node specifications.

Select Yes if you want the configurator to use a naming service.

1.12.1.2 Creating Tower Sets Without Using the Naming Services

If you do not want the configurators to use DECdns or the local namespace, answer No when asked if you want to use a naming service. You will then be asked to specify NSAP or Phase IV addresses instead of node names.

1.12.1.3 Result

The DECNIS text-based configurator will construct the relevant tower sets, and will create entries in the master NCL script to add them to the Known Towers database.

1.12.2 Entering Tower Sets Using the clearVISN DECNIS Configurator

The clearVISN DECNIS configurator only creates tower sets using DECnet Phase IV or NSAP addresses that you supply. However, it has a facility that lets you construct a private database mapping node names to addresses. Once you have set up the node name/address mappings, you can enter node names where you would otherwise enter addresses. Refer to the manual *clearVISN DECNIS Configurator User Guide* for details.

1.12.3 Entering Tower Sets in the Known Towers Database Using NCL Commands

Use the following command to add node names to the Known Towers database:

```
NCL> CREATE NODE decnis SESSION CONTROL -
_NCL> KNOWN TOWER node-name TOWERS = {(tower1), (tower2), ...}
```

where:

decnis is the name of the DECNIS.

node-name is the name by which the remote node is known in the network.

tower1
tower2 are tower sets that describe the protocols and NSAP addresses used to communicate with the remote node.

You must enter one tower for each of the remote node's NSAP addresses.

1.12.4 Tower Set Structure

Each tower set has the following form:

```
([DNA_CMIP-MICE], [DNA_SESSIONCONTROLVn, NUMBER = 19],
[DNA_NSP], [DNA_OSINETWORK, NSAP-address])
```

n is 3 if the tower set refers to a DECnet/OSI node.
is 2 if the tower set refers to a Phase IV node.

NSAP-address is the NSAP address of the node to which the tower set refers. You can specify the NSAP address in Digital format or in OSI format (see Section 16.8).

1.12.5 Example: Adding a Tower Set Using NCL

You want to add the node name ORG:.NORTH.MANCHESTER.SYSTEM2 to the Known Towers database. Assume that this is a DECnet/OSI node with two NSAP addresses as follows:

37:12345:02-00:AA-02-14-78-66-11:20
37:12345:02-00:AA-02-14-78-66-10:20

Enter the following command to place this node in the Known Towers database:

```
NCL> CREATE NODE decnis SESSION CONTROL -
_NCL> KNOWN TOWER org:.north.manchester.system2 TOWERS = { -
_NCL> ([DNA_CMIP-MICE], [DNA_SESSIONCONTROLV3, NUMBER = 19], -
_NCL> [DNA_NSP], [DNA_OSINETWORK, 37:12345:02-00:AA-02-14-78-66-11:20]), -
_NCL> ([DNA_CMIP-MICE], [DNA_SESSIONCONTROLV3, NUMBER = 19], -
_NCL> [DNA_NSP], [DNA_OSINETWORK, 37:12345:02-00:AA-02-14-78-66-10:20])}
```

where *decnis* is the name of the DECNIS.

1.13 DECNIS Trace Facility

DECNIS Trace Facility (DTF) is a utility supplied with the DECNIS software. DTF traces packets as they traverse through the protocol layers within a router. DTF is an enhancement of Digital's CTF (Common Trace Facility) provided in DECnet/OSI; DTF supports multiple platforms and TCP/IP networks.

Note that on Windows 95/NT PCs, you run DTF from an MS–DOS window.

For more information about DTF, refer to the document, DTF.TXT. You can find this in any of the following locations:

Load Host	Location
OpenVMS	SYS$EXAMPLES:DTF.TXT
MS–DOS PCs	*install-dir*\ DTF.TXT
Digital UNIX	usr/lib/dnet/dtf/files/dtf.txt

1.13.1 Polycenter Manager on Netview

The DECNIS can be managed using the POLYCENTER™ Manager on Netview® and the POLYCENTER DECnet Manager. POLYCENTER is a comprehensive system of integrated network management software.

POLYCENTER Manager on Netview lets you control the DECNIS using SNMP management.

The POLYCENTER DECnet Manager lets you monitor, manipulate and test the modules and entities of the management module in the same way as NCL, but it uses a DECwindows interface and provides graphical representations of the status of the system. You can enter the commands described in this manual using POLYCENTER DECnet Manager instead of NCL, either by entering the commands directly or by using the DECwindows™ interface.

The POLYCENTER products also provide alarm notification and performance monitoring.

If you have POLYCENTER DECnet Manager installed when you install the DECNIS software, the installation will customize the Iconic Map window to add the load-host configurator and the DECNIS text-based configurator to the Applications menu.

Refer to the POLYCENTER documentation for more details.

2

Using a Console Terminal on the DECNIS

2.1 Introduction

The DECNIS provides a modem-controlled asynchronous console port on the DECNIS management processor cards MPC-II or MPC-III. The console port provides the following features:

- Support for a VT100-compatible console terminal
- Full-duplex DEC Standard 052 modem control
- An implementation of the Network Control Language (NCL), for managing local and remote nodes from the console terminal
- A console break-in facility
- Password protection

2.1.1 Console Port Location

The console port is situated on the Management Processor Card MPC-II or MPC-III.

2.1.2 Requirements

The following table gives the items required to connect a terminal to the DECNIS console port, and use the console.

Item	Used to...
RS232 cable/interface	Connect the terminal to the console port
DECNIS MPC-II or MPC-III processor card	Provide the console port
VT100-compatible terminal or an MS–DOS PC running a VT100 terminal emulator program	Provide the console

2.2 Starting the Console

Table 2–1 shows how to start a console session at a terminal physically connected to the DECNIS. Refer to Section 2.4.4 to find out how to start a Telnet console session.

If you start the console at a physically connected terminal, then autobaud recognition (autobauding) is always enabled the first time you start the console. Once you have started the console, you can disable autobauding; see Section 2.7.1.

Table 2–1 Starting the Console from a Physically Connected Terminal

If...	Then do this...
Autobaud is enabled	1. Press Return twice. 2. When the character # is displayed, press Ctrl/C. 3. If # is not displayed immediately, repeat steps 1 and 2.
Autobaud is disabled	Press Ctrl/C.

2.2.1 Console Prompt

When the console is started, it will display a copyright screen and the console prompt. The console prompt will be one of the following:

- The DECNIS node name. For example:

  ```
  nis100>
  ```

- If a node name has not been set, the prompt is:

  ```
  console>
  ```

2.3 Exiting from the Console

To exit from the console, do either of the following:

- Enter the following command:

  ```
  console> Exit
  ```

- Press Ctrl/C

2.4 Using Telnet to Set Up a Remote Connection to the Console

You can use the the Telnet protocol to connect to the DECNIS console from a remote terminal. This section describes how to do this.

2.4.1 Introduction

The Telnet protocol is defined in RFC 854. It is a duplex communications facility that allows a standard interface between terminal devices and terminal processes.

The Telnet protocol runs over TCP.

Figure 2–1 shows a terminal using Telnet to connect to the DECNIS. The terminal system is the **Telnet client**; the DECNIS is the **Telnet server**.

2.4.1.1 DECNIS Telnet Implementation

The DECNIS always acts as the Telnet server.

The Telnet client software is not installed on the DECNIS. This prevents the local DECNIS console from being used to set up a remote connection via Telnet to another DECNIS.

2.4.2 Requirements for the Remote Terminal System

The remote terminal system must have:

- A Telnet client implementation installed.
- An IP network connection to the DECNIS.

2.4.3 Setting Up Telnet on the DECNIS

Table 2–2 shows the tasks you must complete on the DECNIS to enable a remote terminal to connect to it using Telnet.

Figure 2–1 Use of Telnet for Remote Connection to DECNIS

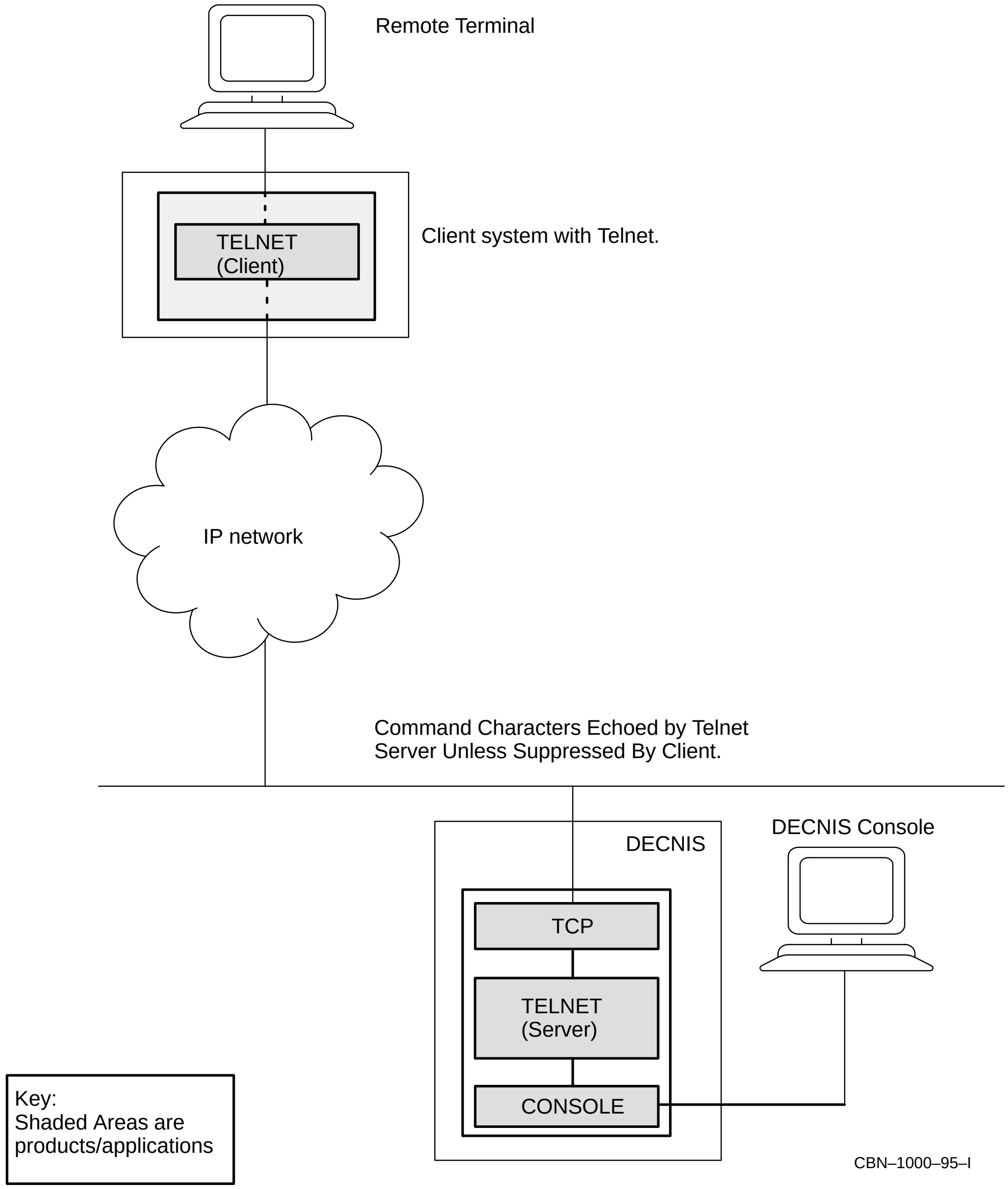


Table 2–2 Tasks Required Before Establishing a Telnet Connection

Task	Action
Enable IP routing	Run the DECNIS configurator (text-based or Windows) and request IP routing.

Create and enable TCP	Run the DECNIS configurator (text-based or Windows); this automatically sets up TCP. Alternatively, issue the following NCL commands:
	`NCL> CREATE TCP`
	`NCL> ENABLE TCP`

If Telnet has been disabled, enable it. Note that Telnet is enabled by default. However, if Telnet connections have been disabled, you must re-enable them.	From the DECNIS console, enter the following command: `console> set telnet on`

Refer to Chapter 1 for more information about the configurators.

2.4.4 Connecting to the DECNIS Console

To create a Telnet connection to the DECNIS console, follow these steps:

Step	Task	Action
1.	From the remote terminal, start Telnet	Issue the appropriate command. For example, on a Digital UNIX system, enter: `# telnet`
2.	From the remote terminal, connect to the DECNIS console:	`telnet> CONNECT` *`id`* where *id* is the IP node name of the DECNIS or one of its IP addresses.
3.	Set up the character echoing option	See Section 2.4.5.

Result of Connecting to the Console

Once connected to the console, you will be asked for a password. Do one of the following:

- If you have set up a console password, as described in Section 2.10, enter this password.
- If you have not set up a console password, enter the DECNIS network management password. Refer to Section 1.7 for details.

When you have entered a password, you will see the console prompt.

2.4.5 Character Echoing Options

You can choose whether characters entered at the remote terminal are echoed locally or remotely. Remote echoing is done by the DECNIS; local echoing is done by the terminal.

2.4.5.1 Remote Echoing

It is recommended that you choose remote echoing. This offers the following advantages:

- Password characters are not echoed, as the DECNIS Telnet server can to determine what should be echoed in the current command context.
- Keyboard command control sequences and responses are processed without you having to press carriage return.

Remote echoing is sometimes referred to as character mode.

2.4.5.2 Local Echoing

It may be preferable to use local echoing if the period between entry and display of a typed character is unacceptable (for example, if the network communications are slow).

If characters are echoed locally, only commands ending with a carriage return are sent to the DECNIS console. For example, in local echoing mode, if you press Up Arrow to recall the previous command line, you then have to press carriage return to send it to the server and get the response.

Local echoing is sometimes referred to as line mode.

2.4.5.3 Procedure for Selecting Echoing Options

The commands needed to select an echoing option will vary according to the remote terminal’s operating system.

Example: OpenVMS

To select remote echoing at an OpenVMS terminal, enter the following command:

```
telnet> SET MODE CHAR
```

To select local echoing at an OpenVMS terminal, enter the following command:

```
telnet> SET MODE LINE
```

2.4.5.4 Returning to the Telnet Prompt from the Console Prompt

To return to the Telnet prompt from the console prompt, do the following:

- If you are using remote echoing, type CTRL/].
- If you are using local echoing, type CTRL/] and then press Return.

2.4.6 Disabling Telnet Connections

To disable all Telnet connections to the DECNIS, enter the following command from the DECNIS console:

```
console> Set Telnet off
```

2.4.7 Specifying Allowed Connections

By default, all Telnet connections to the console are allowed. However, you can specify a list of allowed Telnet sources. If you do this, only the systems you specify will be able to Telnet to the DECNIS console.

2.4.7.1 Procedure

To specify a list of allowed terminals, enter the following command at the DECNIS console:

```
console>  Set telnet sources ip-address ...
```

where *ip-address* is either an actual IP address or an IP address with wildcard elements. The valid IP address formats are shown below:

This IP address format...	Allows these IP addresses to connect...
*	Any
130.223.14.2	130.223.14.2
130.223*	130.223.*xxx.yyy* where *xxx* and *yyy* are numbers in an IP address.
*.223.14.2	*xxx*.233.14.2 where *xxx* is a number in an IP address.

2.4.7.2 Changing Allowed IP Addresses

If you want to change allowed IP addresses, you must reenter all of the information. This is because the current `set telnet sources` command deletes the previous command details.

2.4.7.3 Showing Allowed Telnet Connections

To view the allowed Telnet connections, enter the following command at the DECNIS console prompt:

```
console> Show telnet
```

2.5 Using Console Commands

You can enter console commands to do the following tasks:

Task	Refer to...
Control the console	Section 2.7 to Section 2.10
Control Telnet connections	Section 2.4.3, Section 2.4.6 to Section 2.4.7.3
Do an IP Domain Name Server lookup	Section 2.11
Control the contents of flash memory	Chapter 3
Load and dump the DECNIS	Section 2.12
Start NCL	Section 2.13
Use console break-in	Section 2.14 to Section 2.15

2.5.1 Displaying Console Command Descriptions

To see a list of console commands, enter the following command:

```
console> help
```

2.5.2 Help with Console Command Parameters

To find out the valid parameters for a command, press ? and Return after a command verb or parameter.

Example

If you enter:

```
console> set ?
```

The screen displays all the attributes of the `set` command.

If you enter:

```
console> set console ?
```

The screen displays all the attributes of the `set console` command.

2.5.3 Console Command Summary

Table 2–3 lists the console commands.

Table 2–3 Console Command Summary

Command	Description	
Add *script* *node*:*filename*	Adds a configuration script to flash memory. Refer to Chapter 3	
Boot	Reruns the system self-test and boots the DECNIS	
Cls	Clears the screen	
Delete *n* where *n* is a flash index number	Marks a file in flash memory as unusable. Refer to Chapter 3	
Disconnect	Disconnects from the modem and exits	
Dump	Dumps the DECNIS and reruns the system self-test	
Exit	Exits the console	
Help	Displays help text	
Load...	Loads the DECNIS using the method specified	
Lookup	Performs an IP Domain Name Server lookup.	
Ncl	Starts the Network Control Language (NCL) utility	
Password	Sets a new console password	
Restart	Reloads the DECNIS using its default method	
Set console...	Sets console attributes, as follows:	
	Set console [no]autobaud	Disables or enables console autobauding
	Set console [no]disconnect	Disables or enables disconnecting the modem on console exit
	Set console [no]modem	Disables or enables modem control on the console port
	Set console speed	Sets the console speed
	Set console timeout	Sets the default inactivity timeout period for console sessions using a physically connected terminal

(continued on next page)

Table 2–3 (Cont.) Console Command Summary

Command	Description	
Set flash...	Sets flash memory attributes, as follows:	
	Set flash boot *n* (only valid for MPC-III cards)	Specifies the area of flash memory from which the DECNIS should be loaded.
	Set flash image *n*	Specifies the software image to be loaded from flash memory
	Set flash script *n*	Specifies the configuration script to be loaded from flash memory
	Set flash update *n* (only valid for MPC-III cards)	Specifies the area of flash memory to be cleared and updated with a new image or combined image.
Set session...	Sets attributes for the session, as follows:	
	Set session size	Sets the screen size for the session
	Set session timeout	Sets the inactivity timeout period for the current session
	Set session username	Sets the username for the session
Set Telnet...	Sets Telnet attributes, as follows:	
	Set Telnet off	Disables Telnet connections
	Set Telnet on	Enables Telnet connections
	Set Telnet sources	Sets up permitted sources for Telnet connections
	Set Telnet timeout	Sets the default inactivity timeout period for Telnet sessions
Show console	Shows current console attributes	
Show flash	Shows current flash memory attributes and contents	
Show session	Shows current session attributes	
Show Telnet	Shows current Telnet attributes	
Update *node*:*file-name*	Updates with the specified file the area of flash memory specified with set update	
Users	Displays the current users of the console	

2.6 Editing Console Commands

Table 2–4 lists the keys used to edit console commands, and recall previous commands. Note that editing is always in overstrike mode.

Table 2–4 Keys for Editing Console Commands

Use this key...	To do this...
Up arrow	Recall up to ten previous commands.
Down arrow	Re-examine command previously recalled
Right arrow	Move the cursor one character to the right
Left arrow	Move the cursor one character to the left
Ctrl/R	Redraw current line
⌫ or DEL	Delete the character to the left of the cursor
Ctrl/U	Delete text from the cursor position to the start of the line

Table 2–5 lists the keys used to control command output.

Table 2–5 Keys to Control Command Output

Use this key...	To do this...
Ctrl/C	Cancel the current command or output
Ctrl/O	Suspend or continue display of output to the terminal
Ctrl/X	Cancel the current line, and delete data that has been typed but not yet displayed or executed

2.7 Disabling Autobauding and Setting the Console Speed

By default, the console enables autobauding when you start the console from a physically connected terminal. This means that the console automatically tries to set the console port speed to be compatible with the terminal port speed.

If your terminal has a fixed port speed, you may want to turn off autobauding, and set the console port speed yourself. Section 2.7.1 describes how to do this.

If you turn off autobauding without setting a new port speed, then the next time you start the console, it will set it to the last speed used.

Note that autobauding does not apply to Telnet sessions.

2.7.1 Procedure: Disabling Autobauding

Follow these steps:

1. To turn off autobauding, enter the following command:

   ```
   console> set console noautobaud
   ```

2. To set the console port speed, enter the following command:

   ```
   console> set console speed n
   ```

   where *n* is one of the following speeds, in kbits/s:

   300, 600, 1200, 2400, 4800, 9600, 19200

3. If the new speed is different from the one currently used, you are asked to confirm the change. Enter Yes.

Result: The new console speed will take effect immediately.

2.7.2 Autobauding Restored When DECNIS Is Restarted

If you power up the DECNIS with the dump button pressed in, then the next time you use the console, it will use autobauding.

2.8 Setting Inactivity Timeout Periods

The console will automatically exit if there has been no console activity for a given period. This period is known as the **inactivity timeout period**. Console activity means either that you have typed something, or that the console has displayed something.

You can change inactivity timeout periods, or disable inactivity timeout altogether. Table 2–6 shows the different types of console inactivity timeout periods that you can specify.

Default Inactivity Timeout Period

All inactivity timeout periods are by default set to 300 seconds.

2.8.1 Procedure

To change the inactivity timeout period, enter the appropriate command as shown in Table 2–6:

Table 2–6 Commands to Change Inactivity Timeout

To do this...	Enter this command
Set the default inactivity timeout for sessions using a physically connected console terminal.	`console> Set console timeout` *n*
Set the default inactivity timeout for Telnet sessions	`console> Set telnet timeout` *n*
Set the inactivity timeout for the current console session (Telnet or physically connected)	`console> Set session timeout` *n*

where *n* is a decimal number in the range 10 to 65535. It specifies the number of seconds that the console is allowed to be inactive before it exits.

2.8.1.1 Disabling Inactivity Timeout

If you do not want console sessions to exit automatically after an inactive period, make the inactivity timeout period zero (0). For example, for Telnet console sessions, enter:

```
console> Set Telnet timeout 0
```

2.9 Disabling and Enabling the Modem

By default, modem control is disabled. If your terminal or PC does not have a modem, or its modem will work without console modem control, then leave modem control disabled.

However, If your terminal or PC has a modem or a null modem cable, you may enable modem control.

Note that the modem control does not apply to Telnet sessions.

2.9.1 Procedure

- To enable modem control, enter the following command:

  ```
  console> set console modem
  ```

- To disable modem control again, enter the following command:

  ```
  console> set console nomodem
  ```

2.9.2 Automatic Modem Disconnection

By default, the console port modem is automatically disconnected when the console exits. If you want the modem to remain connected when the console exits, enter the following command:

```
console> set console nodisconnect
```

2.10 Setting and Disabling a Console Password

You can limit access to the console by setting a console password. You can also disable a current password.

2.10.1 Procedure: Setting a Password

To set or change a console password, enter this command:

```
console> password
```

You will then be asked to enter the new password and to reenter it, for verification. If a console password already exists, you are asked for the current password before being allowed to change it.

Syntax

The password is an alphanumeric string of up to 8 characters.

2.10.2 Procedure: Disabling a Password

To disable an existing password, follow these steps:

1. Enter this command:

   ```
   console> password
   ```

2. Enter the current password when prompted.
3. Press Return at the prompts for `New Password` and `Verification`.

Result: You will not be asked for a password when you access the console.

2.10.3 Telnet Password Requirement

Whether or not you have set up a password, you are still required to enter a password if you connect to the console using Telnet.

If no password has been explicitly set up, the default password is the network management password of the DECNIS. Refer to Section 1.7 and Section 6.3 for more information about this password.

2.11 Looking Up IP Addresses and Node Names

You can use the lookup command to find the IP address corresponding to an IP node name, or vice versa.

2.11.1 Requirement

Before you can issue the lookup command, you must have set up the IP Domain Name Server by entering the following NCL commands

```
NCL> CREATE IP SERVICES RESOLVER
NCL> CREATE IP SERVICES RESOLVER SERVER server-name IP ADDRESS ip-address
NCL> ENABLE IP SERVICES RESOLVER
```

where *server-name* is the name of an IP Domain Name Server willing to answer queries, and *IP-address* is its IP address.

2.11.2 Procedure

To do a Domain Name Server lookup, enter the following command:

```
console> lookup node-spec
```

where *node-spec* is either an IP address or an IP node name.

Result: If you enter an IP address, the console displays the corresponding IP node name. If you enter an IP node name, the console displays the corresponding IP address.

2.12 Loading and Dumping the DECNIS

This section describes the commands used to control loading, rebooting and dumping of the DECNIS.

2.12.1 Booting the DECNIS

To reboot the DECNIS, follow these steps:

1. Enter the following command:

   ```
   console> Boot
   ```

2. You are asked to confirm that you want to go on. Enter Yes.

Result: The DECNIS will rerun the system self-test. It will then reload, using its default type of loading. This is either of the following:

- On a Windows 95/NT load host, the type of loading specified in the clearVISN DECNIS configurator.

- On any other load host, the default is the type of loading specified in the load-host configurator.

2.12.2 Dumping the DECNIS

To force the DECNIS to dump, follow these steps:

1. Enter the following command:

   ```
   console> Dump
   ```

2. You are asked to confirm that you want to go on. Enter Yes.

Result: The DECNIS will dump to a dump host. Then, once the dump completes (or times out), it will run its self-test. Finally, it will reload, using its default type of loading (as defined in Section 2.12.1).

2.12.3 Loading the DECNIS

To reload a DECNIS, and specify the type of loading, enter the following command:

```
console> load loadtype
```

where *loadtype* is one of the following:

- –FLASH to reload the image and (if present) the configuration file from flash memory.

  If there is more than one image and/or configuration file in flash memory, you can specify which one will be loaded; refer to Chapter 3 for details.

- –NETWORK to reload the software image and configuration files from a load host.
- A file name to load a specified file from a load host. Do not specify a directory or pathname.

Specifying a File Name

If you specify a file in the load command, be sure that there is a file of that name in the correct directory on the load host.

2.12.4 Restarting the DECNIS

The restart command has the same effect as the load command, except that you cannot specify the type of loading or a load file. The restart command always uses the default type of loading (as defined in Section 2.12.1).

Procedure

To restart the DECNIS, enter the following command:

```
console> restart
```

2.13 Using the Network Control Language

You can run the Network Control Language (NCL) on the console. Refer to Section 1.3 for a general description of NCL.

Note that only one user at a time can run NCL during a DECNIS console session. If a second user attempts to use NCL, a message appears asking if the current user should be logged off.

2.13.1 Starting NCL

To start NCL on the console, enter the following command:

```
console> ncl
```

2.13.2 Setting the Terminal Screen Size

Each time you start NCL, the NCL program will try to determine the screen size, using ANSI escape sequences. You can override this automatic sizing by specifying the screen size to be used during the current console session.

2.13.2.1 Procedure

To set the terminal screen size for the current session, enter the following command:

```
console> set session size row column
```

where *row* is the number of rows on the screen and *column* is the number of columns on the screen.

Example

To specify a screen size of 24 rows by 80 columns, enter the following command:

```
console> set session size 24 80
```

2.13.3 Exiting from NCL

To exit from NCL on the console, type any of the following at the NCL prompt:

- Ctrl/Z
- Exit and Return

You will return to the console prompt.

2.13.4 NCL Command Format

2.13.4.1 Managing the Local DECNIS

When using NCL to manage the local DECNIS, issue the NCL command in the following format:

```
verb entity-name [{argument(s)/attribute(s)}]
```

2.13.4.2 Managing a Remote Node

When using NCL to manage a remote node, specify the node in the NCL command, as follows:

```
verb NODE node entity-name [{argument(s)/attribute(s)}]
```

where *node* is one of the node specifications listed in Table 2–7.

2.13.4.3 Supplying Access Control Information

To supply access control information in console NCL commands, use the following format:

```
verb NODE node/username/password  entity-name [{argument(s)/attribute(s)}]
```

2.13.4.4 Using Default Nodes

If you wish to issue several NCL commands to the same remote node, you can set up that node as the default by entering the following command:

```
NCL> SET NCL DEFAULT ENTITY NODE node/username/password
```

You can then enter NCL commands without specifying the node name.

2.13.5 Setting Default Security

If you have set default access to NCL on a remote node, you can set default security for that node by issuing the following NCL command:

```
NCL> SET NCL DEFAULT ACCESS BY USER username, PASSWORD password
```

2.13.6 Managing the DECNIS and Other Nodes

You can use console NCL to manage any node that is normally manageable by NCL over a DECnet network, for example, DECNIS, DECnet/OSI or WANrouter 90 nodes.

In addition, if you set up the TCP entity on the DECNIS, as described in Section 1.9, you can manage the following types of system, whether or not they are running DECnet:

- Digital UNIX V3.0 or later systems
- SNA Gateway V2.0 or later systems

2.13.6.1 Specifying the Node

Table 2–7 shows the syntax for specifying a remote node in console NCL commands.

Table 2–7 Node Specifications in NCL commands

Node Specification	Example Syntax	See Also
DECnet Phase IV address	`NCL>SHOW NODE 1.100`	
DECnet Node name	`NCL>SHOW NODE nis100`	Section 2.13.6.2 and Section 2.13.6.5
IP address	`SHOW NODE 16.36.16.100`	Section 2.13.6.3
IP node name	`SHOW NODE nis100`	Section 2.13.6.4

2.13.6.2 Using DECnet Node Names in NCL Commands

To use a DECnet node name to identify a node in a console NCL command, you must do the following:

1. In the CREATE user NCL script file for the DECNIS, enter NCL commands that create a SESSION CONTROL KNOWN TOWER entity for each node you expect to manage from the DECNIS. Refer to Section 1.5.6 and Section 1.12.3 for more information.
2. Use the name of the SESSION CONTROL KNOWN TOWER entity for the node as the node name.

Example

This is an example of an NCL command to create a SESSION CONTROL KNOWN TOWER entity for the node NIS100.

```
CREATE NODE NIS100 SESSION CONTROL KNOWN TOWER -
 AA-00-04-00-DE-11-A0-AA-F9-6F-56-DE-8E-00:.nis100 TOWERS -
  {([ DNA_CMIP-MICE ], [ DNA_SessionControlV2 , Number = 25 ], -
   [ DNA_NSP ], [ DNA_OSINetwork , 49::00-01:aa-00-04-00-64-04:20 ])}
```

2.13.6.3 Using IP Addresses in NCL Commands

In order for you to use an IP address to identify a node in a console NCL command, the TCP entity must have been created and enabled on the DECNIS. This is done automatically by the DECNIS configurators.

2.13.6.4 Using IP Node Names in NCL Commands

To use an IP node name to identify a node in a console NCL command, you must follow all the steps in Section 1.9.3.

2.13.6.5 How the DECNIS Interprets Node Names and Addresses

Refer to Section 1.9.3.1 for details.

2.13.7 Help on NCL

The online help within NCL gives details of the commands, entities, attributes and characteristics supported. Later chapters in this manual give details of the NCL commands used to carry out specific tasks.

Starting NCL Help

To get NCL help on the DECNIS, follow these steps:

1. Call up the online help by entering HELP at the NCL prompt:

   ```
   ncl> HELP
   ```

2. Select the menu item `DECNIS`.

Exiting from NCL Help

To exit from NCL help, type Ctrl/D or Ctrl/Z.

2.13.8 Editing NCL Commands

Table 2–8 lists the keys used to edit NCL commands, and recall previous commands.

Table 2–8 Keys for Editing NCL Commands

Use this key...	To do this...
Up arrow	Recall a previous command
Down arrow	After recalling a previous command, recall the next command in the series
Left arrow	Move the cursor one character to the left
Right arrow	Move the cursor one character to the right
Enter or Return	Enter a command you have typed
F14 or Insert	Toggle between inserting characters and typing over characters
F12 or Ctrl/A	Move the cursor to the beginning of the line

(continued on next page)

Table 2–8 (Cont.) Keys for Editing NCL Commands

Use this key...	To do this...
Ctrl/E	Move the cursor to the end of the line
Ctrl/R	Redraw current line
Ctrl/U	Delete text from the cursor to the beginning of the line
⌫ or DEL	Delete the character to the left of the cursor
F13 (not on all keyboards)	Delete the word to the left of the cursor
-	Type a hyphen at the end of a line to continue typing a command on the next line
^ *string* Return or *string* Find Return	Recall the last NCL command that starts with the characters in *string*

2.13.8.1 Line Length

The maximum length of a line in an NCL command is 1024 characters. The effect of this is as follows:

- If you are typing in an NCL command, when you reach the righthand side of the screen, the cursor automatically moves to the next line.
- If you have typed 1024 characters or less, you can at any point go back to previous lines (using the left arrow key) and edit the command.
- Once you type more than 1024 characters, the cursor jumps to the next line. At this stage, you cannot go back and edit previous lines. You must press Enter to enter the command.

2.13.8.2 Command Length

The maximum length of an NCL command (as opposed to a line) is 2048 characters. To type in a command longer than 1024 characters, use a hyphen as a continuation character.

You can enter the hyphen at any point before you have typed 1024 characters.

2.13.9 Displaying NCL Output

You use NCL SHOW commands to display information about your system. The console provides a UNIX-style **More** function to control the display of NCL command output.

2.13.9.1 Controlling the Display of NCL Output

When you enter an NCL SHOW command, the console displays the first screenful of NCL output. Table 2–9 shows the keys you can then use to control the display of NCL command output.

Table 2–9 Keys to Control NCL Output

Use this key...	To do this...
Space bar	Display the next screen of NCL output.
Return	Scroll one line of output.
q or Q	Terminate the output and return to the NCL prompt.
b or Ctrl/B	Display the previous screen of output.
? or h	Display help text on the More utility.
s or S	Scroll continuously to the end of the output.

2.13.9.2 The More Prompt

When you enter an NCL SHOW command, the console displays the following prompt at the bottom of the screen:

```
--More--(x)
```

where *x* is the screen number of this screen of output.

The screen number is relative to the other screens of output for this command. For example, if an NCL command produces three screens of output, and you are looking at the second screen, the prompt is:

```
--More--(2)
```

If you page back to a previous screen of output, the prompt will contain an arrow. For example, if you page back to the first screen of output, the prompt will be:

```
--More->(1)
```

If you page back until you are looking at the output from a previous command, the prompt will show a negative number. For example:

```
--More->(-3)
```

When the last saved screen is displayed, you will see the prompt:

```
No more screens saved
```

2.13.10 Restrictions

This section describes restrictions in the use of NCL at the console.

2.13.10.1 SNAPSHOT Command

The NCL SNAPSHOT command is not supported in console NCL.

2.13.10.2 NCL File Generation

You cannot create or use NCL script files, or generate log files from the console.

2.13.10.3 DECdns Names in NCL Commands

A restriction applies to NCL commands containing attributes which are DECdns names. If you enter an NCL command of this type at the console, the DECdns name must have the following format:

```
NSCTS:.dns-name
```

where: *NSCTS* is the namespace creation timestamp of the namespace in which the full name is registered.

dnsname is the DECdns full name of the node.

Definition

The NSCTS is a unique identifier for the namespace in which the DECdns full name is registered. The NSCTS is automatically created when the namespace is created. It consists of 14 hexadecimal digits.

Finding the NSCTS

There are two ways to find the NSCTS for a node. These are described in the following sections.

Finding the NSCTS: Using the Phase IV Address

If the node has a DECnet Phase IV address, follow these steps:

1. Find the DECnet Phase IV address of the node.
2. Enter the following NCL command at the console:

   ```
   ncl> SHOW NODE phase-iv-address NAME
   ```

   where *phase-iv-address* is the Phase IV address of the node.
3. In the display, the hexadecimal digits appearing before the colon (:) make up the NSCTS.

Finding the NSCTS: Using the Node Name

If the node does not have a DECnet Phase IV address, follow these steps:

1. From a DECnet/OSI node, enter the following NCL command:

   ```
   ncl> SHOW NODE node-name DNS CLERK KNOWN NAMESPACE * NSCTS
   ```

   where *node-name* is the DNS name of the node whose NSCTS you are attempting to find.

2. The NSCTS is displayed next to the label `NSCTS`.

Example Commands

This section gives example NCL commands to which this restriction applies. In the examples:

- The node name to be used as an attribute is:

  shrub.leaf

- The NSCTS is:

  00-12-23-56-77-A0-A1-A2-A3-A4-A5-A6-A7-18

Example 1

To define **shrub.leaf** as an event sink for a router called PEACH, enter this command:

```
ncl> SET NODE peach EVENT DISPATCHER SINK NODE -
_ncl> {00-12-23-56-77-A0-A1-A2-A3-A4-A5-A6-A7-18:.shrub.leaf}
```

Example 2

To set up X.25 security for outgoing calls for the node **shrub.leaf**, enter this command:

```
ncl> SET X25 SERVER SECURITY NODES security_out NODES -
_ncl> {00-12-23-56-77-A0-A1-A2-A3-A4-A5-A6-A7-18:.shrub.leaf}
```

2.14 Using Console Break-in

The console provides an emergency break-in function. This allows you to use a limited set of commands even if the DECNIS is not responding to normal console commands, or commands over the network.

2.14.1 Procedure

To use the break-in function, follow the steps in the following table:

If...	Then do this at the console...
The DECNIS is loading or dumping	1. Press Ctrl/P and then press Return. 2. The ROM console program will start; see Section 2.15 for details.
The DECNIS is not loading or dumping	1. Press Ctrl/P. 2. If a console password is set, you are asked to enter it. If no password is set, you go directly to step 3. 3. You are prompted to enter a single letter of the break-in command to be executed: `Choose one [D]ump, [H]alt, [L]oad, [Q]uit ?`

2.14.2 The Break-In Commands

Table 2–10 describes the break-in commands.

Table 2–10 Break-in Commands

Command	Description
Dump	Dump the DECNIS. This is the same as the console command **dump**.
Halt	Stops the DECNIS and the normal console software. Refer to Section 2.15.
Load	Reload the DECNIS. This is the same as the console command **restart**.
Quit	Return to the normal console prompt.

Note that normal console output is suspended while the break-in prompts are active.

2.15 The ROM Console Program

The DECNIS provides a ROM-based console program which you can access even if the DECNIS is not loaded or the software console program has stopped. The ROM console program provides a limited set of console commands, which you can use for diagnostic purposes or to reload the DECNIS.

2.15.1 Starting the ROM Console Program

The ROM console program starts if you do either of the following:

- Select Halt when using the break-in function.
- Press Ctrl/P and then Return when loading or dumping the DECNIS.

2.15.2 ROM Console Commands

Table 2–11 lists the ROM console commands.

Table 2–11 ROM Console Command Summary

Command	Description
Boot	As in Table 2–3
Dump	Dumps one or more areas of DECNIS memory
Help	Displays a list of ROM console commands
Load	As in Table 2–3
Set console modem	Sets the modem signal power up state, and enables or disables modem control
Set console speed	As in Table 2–3
Set flash boot	As in Table 2–3
Set flash image	As in Table 2–3
Set flash script	As in Table 2–3
Set flash update	As in Table 2–3
Set self_test	Specifies a quick or a full self-test
Show all	Shows console parameters
Show console	As in Table 2–3
Show flash	As in Table 2–3
Show memory	Show memory size
Show self_test	Shows the type of self-test set
Zero [BBRAM]	Empties DECNIS BBRAM (memory in battery-backed RAM). Note that this command does not empty flash memory.

2.15.3 Dumping the DECNIS

To force the DECNIS to dump, enter the following command:

```
dump dumptype
```

where *dumptype* can have any of the following values:

One or more of the following:	
MPC	Dumps memory on processor card
POOL	Dumps pool memory on memory card
CARDS	Dumps Network Interface Card memory
ARE	Dumps ARE memory on memory card
DEFAULT or no parameter	Dumps the DECNIS memory
NONE	Dumps a minimum area of memory. Use this to test that the dump command works.

Example

Enter the following to dump MPC memory and pool memory:

```
dump {mpc, pool}
```

Recommendation

It is strongly recommended that you enter the dump command without parameters, to create a full dump. Normally, a full dump is required to analyze problems correctly.

2.15.4 Enabling Modem Control

By default, modem control is disabled. To enable modem control from the ROM console, enter the following command:

```
set console modem on
```

For more information, refer to Section 2.9.

2.15.5 Enabling Modem Control

To set the state of the modem signals when the DECNIS powers up, enter the following command:

```
set console modem signaltype
```

where *signaltype* can have any of the following values:

DTR
RTS
DSRS

ALL

2.15.6 Set and Show Self-Test

You can choose whether you want the DECNIS to run a full or a quick self-test on starting up.

- For a full self-test, enter this command:

  ```
  set self_test full
  ```

  The full self-test takes about five minutes.

- For quick self-test, enter this command:

  ```
  set self_test quick
  ```

  The quick self-test takes about 30 seconds.

2.15.6.1 Show All Command

The show all command displays useful information about the console parameters set. For example, it might show the following:

```
Model                               : DECNIS 600-VE
Node id                             : 08-00-2B-A1-12-80
Self-test mode                      : Quick
Console speed                       : 19200
Console modem signal powerup state  : DTR-on   RTS-on   DSRS-off
Secure console mode                 : On
MPC memory size (Mbytes)            : 16
Configured flash RAM size (Mbytes)  :
Image index                         :
Script index                        :
Flash contents                      : Index   Name            Size
                                    : 1       sysncl          2260940
                                    : 2       sysnoncl        1911244
```

3

Configuring Nonvolatile Memory Dynamically

3.1 Introduction

This chapter describes how to modify the contents of nonvolatile (flash) memory after the image file or combined file has been loaded.

- Section 3.2 describes the commands and procedures for controlling the content of nonvolatile memory on the MPC–II. The same commands are also applicable to the MPC–III.
- Section 3.3 describes the additional commands required for working with an MPC–III, and illustrates these commands with a worked example.

3.1.1 Dynamic Updates to Nonvolatile Memory

You can use the DECNIS console to add new CMIP files (scripts) to nonvolatile memory dynamically, and to select which script is to be used at the next reboot.

This feature saves having to reload the DECNIS from a host when testing new configuration scripts.

Management Processor Cards MPC–II and MPC–III

The DECNIS console is only supported on the management processor cards MPC–II and MPC–III. The MPC–II has 4 MB of flash memory and the MPC–III has 8 MB.

You can use the console to dynamically add, delete, update and view files in nonvolatile memory, and designate the file to be loaded. The console commands are the same for either card, but there are some additional considerations for the MPC–III, since on this card flash memory is divided into two distinct areas.

3.1.2 Requirements

The ability to dynamically modify the contents of flash memory is only useful if the DECNIS is configured so that it loads from flash memory. *DECNIS Configuration and Loading* describes how to prepare and configure a load file for the DECNIS, and the basic procedure for setting up the DECNIS to load from flash memory. Chapter 1 also briefly describes these procedures.

3.2 Controlling Nonvolatile Memory on the MPC–II

The commands described in this section also apply to the MPC–III. However, before you can use them on the MPC–III, you need to do some additional configuration; refer to Section 3.3 for details.

3.2.1 Adding a File

You can add script files and profiles to flash memory using the ADD console command, as follows:

```
console> add flash-name host:file-name
```

where:

flash-name is the name to be used for the script or profile file in flash memory.

host is the IP address or name of the remote host.

file-name is the specification of the file to be added.

You can keep adding scripts and profiles until flash memory is full. It may be possible to add as many as 32 different scripts into flash memory, depending on the size of the scripts.

3.2.1.1 Script File Names

If you want to be able to use a script file as the default script, its flash name should be of the form `script` or `script.`*yy*. The extension *yy* can be any string. Script file names do not have to be unique.

3.2.1.2 Profile File Names

The flash name used for a profile file must be the correct name for the profile (`mcnm_prf`, `x25l2_prf` or `x25l3_prf`). Delete any older versions from flash memory and from the host.

3.2.1.3 Remote Host System Requirements

The remote host system is the system where the scripts are held. It need not be the load host used for the load image, but it must meet the following requirements:

- It must have a TFTP server.
- The load files and the directories in which they are held must be accessible by the TFTP server. For example, on OpenVMS and Digital UNIX systems, they must be world readable.

Special Considerations

Refer to the release notes for information about problems found when using particular TFTP servers.

For example, problems have been found with the TFTP server currently provided by UCX software on OpenVMS hosts. The DECNIS release notes describe the problems and solutions.

3.2.2 Selecting the Script or Image to Be Loaded

If there are several configuration scripts in flash memory, you can specify which should be loaded on the next load from flash memory. If you do not specify the script to be loaded, then by default, the first file with the name `script` or `script.`*yy* is loaded.

You specify the script to be loaded by giving its **index number**. The next sections explain how to find the script index number and how to specify the script.

Section 3.2.3.2 explains how to specify the image to be loaded

3.2.2.1 Finding the Flash Index Numbers

Each new file is added to flash memory in sequence and an index number is assigned to it. To find the flash index number enter the command:

```
console> show flash
```

This displays the images and configuration scripts in flash memory. For example:

```
Image Index          0
Script Index         7
Flash Contents       Index   Name            Size      Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:37:32.380
  sumchk ok  32415    2       script          7416      1996-08-27-16:21:57.308
  sumchk ok  42836    3       ospf            5824      1996-08-27-16:34:24.308
  sumchk ok  28463    4       script.1        8420      1996-08-27-16:54:17.367
  sumchk ok  24778    5       mcnm_prf        1028      1996-08-27-17:12:46.335
  sumchk ok  32449    6       x25l2_prf       18650     1996-08-27-17:35:08.153
  sumchk ok  17232    7       x25l3_prf       23826     1996-08-27-17:54:21.302
```

3.2.2.2 Using the Default Script

The default image is the first or only file in the displayed list.

The default script is the first file called `script` or `script.`*yy*, where *yy* can be any string.

For example, you may have two script files named `script`, one with the index number 2 and one with the index number 3. The script file with the index number 2 will be loaded by default.

3.2.3 Selecting the File to Be Loaded from Flash Memory

This section describes how to override the default script by specifying another script to be loaded.

3.2.3.1 Specifying the Script to Be Loaded

To specify a configuration script to be the next script loaded, enter the following command:

```
console> set flash script index-number
```

where *index-number* is the index number of the image or script.

If you want to go back to the default script, enter 0 (zero) for the *index-number*.

Note that if you clear the contents of NVRAM (for example, by holding the dump button in when you power up the DECNIS), this value is reset to the default (0).

Example

In the example display in Section 3.2.2.1, you wish to select the file called `script.1` as the script file to load. Enter the following command:

```
console> set flash script 4
```

The file `script.1` will be loaded when you next reboot the DECNIS.

3.2.3.2 Specifying the Image to Be Loaded

If you have loaded more than one image into flash memory, you can specify which should be loaded next. Although you would not normally load more than one image into flash memory, you may wish to do this for test purposes.

Procedure

The default image is the first image in the displayed list. To specify another image, enter the following command:

```
console> set flash image index-number
```

where *index-number* is the index number of the image. To go back to loading the default image, enter 0 (zero) for the *index-number*.

3.2.3.3 Entering Incorrect Index Numbers

If you specify a nonexistent or inappropriate index number, and then try to load the DECNIS from flash memory, the illegal number will be ignored. The DECNIS will instead try to load the default image or script.

An inappropriate index number is one that is wrong for the type of file, for example, if you specify as the script number the index number of a profile file.

3.2.4 Deleting an Entry

You can remove a script or other file entry from the flash index.

Procedure

To remove a file, enter the command:

```
console> delete index
```

where *index* is the index number of the file being deleted.

This marks the file as deleted. It does not physically remove the file from flash memory, it simply removes the entry for the file from the flash index. The other index entries are not affected.

Example

In the example display in Section 3.2.2.1, you wish to delete the file called `script`. You would enter the following command:

```
console> delete 2
 File index 2 will be marked as deleted,
 but the space cannot be recovered, Confirm ? Y
```

To see the result, use the SHOW FLASH command:

```
console> show flash
Image Index          0
Script Index         7
Flash Contents       Index   Name            Size      Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:37:32.380
  sumchk ok  42836    3       ospf            5824      1996-08-27-16:34:24.308
  sumchk ok  28463    4       script.1        8420      1996-08-27-16:54:17.367
  sumchk ok  24778    5       mcnm_prf        1028      1996-08-27-17:12:46.335
  sumchk ok  32449    6       x25l2_prf       18650     1996-08-27-17:35:08.153
  sumchk ok  17232    7       x25l3_prf       23826     1996-08-27-17:54:21.302
```

Now that `script` has been deleted from the index, `script.1` becomes the default script.

Memory Allocation

Note that although the DELETE command marks a file as deleted and its entry is removed from the index, the space allocated to the file is not recovered.

To recover the allocated space you must delete the file from the combined image file on the load host using the MOD_FLSH command. Then you can clear flash and reload the amended image from the load host using the UPDATE command, as described in Section 3.2.5.

3.2.5 Updating Nonvolatile Memory

You can erase the contents of flash memory and load a new image or combined file into flash memory. To do this, enter the following command:

```
console> update host:file-name
```

where:

host is the IP address or name of the remote host.

file-name is the specification of the file to be loaded.

For example:

```
console> update 16.36.0.138:/usr/users/flash/nis_nis116.sys
 About to overwrite flash area, Confirm ? y
Is it ok to clear the flash first ? y
```

The first question is asking for confirmation that you want to proceed. Answer "Y" to continue, or "N" to abort the command.

The second question is asking if it should clear flash memory first.

If you answer "N" the system will check that the file exists and is of the correct format, before it erases flash memory and starts to load the image. You may need to adjust the TFTP timeout interval because the time taken to erase the flash memory exceeds the default TFTP timeout value. If TFTP times out the connection is lost by the time flash memory is cleared and the transfer fails.

If you answer "Y" the system clears flash memory immediately, before it makes a connection to the load host.

Note that flash memory is cleared before the load starts. If there is a failure during the transfer (for example, if there is a power failure), then the contents of flash memory is lost. If you have an MPC-II card the DECNIS has to load a new image from the load host before it can reboot. If you have an MPC-III card the effect is not so serious because only the Flash Update Area is erased. See Section 3.3 for more information about the use of flash memory on the MPC-III.

3.3 Controlling Nonvolatile Memory On the MPC–III

You can use the same commands on the MPC–III as on the MPC–III. However, before you can use them on the MPC–III, you need to do some additional configuration, as described in the next sections.

3.3.1 Nonvolatile Memory Areas

The MPC–III has 8 MB of flash memory, as against 4 MB on the MPC–II. These 8 MB are divided into two areas, as shown in Figure 3–1.

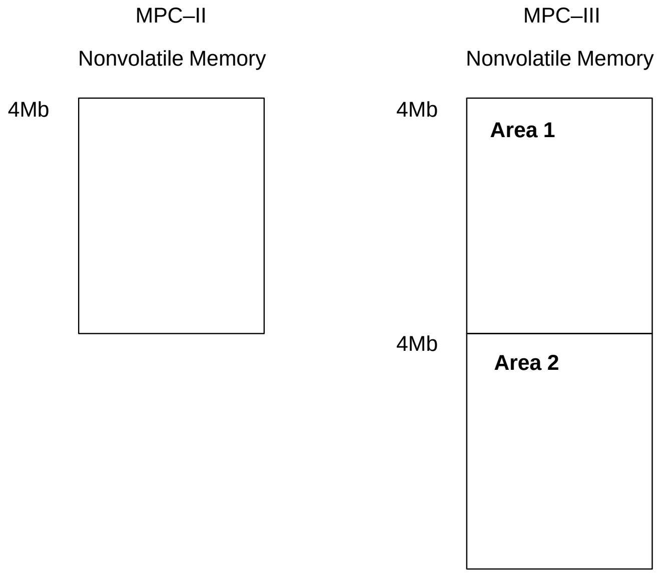
On an MPC–III, you designate the way these areas are to be used, as follows:

- You can designate one of the areas of flash memory to be the Flash Boot Area. This will be the area that the DECNIS will load from at startup.
- If you want to control files in flash memory, you must first designate an area to be used as the Flash Update Area. The commands to add, delete, set or update files only operate in the Flash Update Area.

  Once you have designated the Flash Update Area, you can use the ADD, DELETE, SET and UPDATE commands, as for the MPC–II; refer to Section 3.2.

The Flash Boot Area and Flash Update Area are set independently. They may both refer to the same area of flash memory or to different areas.

Figure 3–1 Flash Memory on MPC–II and MPC–III



CBN–0050–93–I

3.3.2 Selecting the Flash Boot Area

With the MPC–III, the DECNIS boots from the flash memory area which has been designated as the Flash Boot Area. By default, the DECNIS will use flash Area 1 as the Flash Boot Area.

3.3.2.1 Changing the Flash Boot Area

To change the Flash Boot Area, enter the following console command:

```
console> set flash boot area-number
```

where *area-number* is the number of the area which contains the combined image you want the DECNIS to use the next time it is started.

There is an equivalent ROM console command which you can use to set the Flash Boot Area, if an error occurs during loading; refer to Section 2.15.2.

Note that, if the load from the Flash Boot Area fails, the DECNIS does **not** access the other area. If an error occurs in loading the DECNIS attempts to load from a load host.

3.3.3 Selecting the Flash Update Area

By default, the Flash Update Area is not defined.

Commands which affect the contents of flash memory only operate on the area which has been nominated as the Flash Update Area. This means that the ADD, DELETE, SET and UPDATE console commands will not work on the MPC–III until you have selected the Flash Update Area.

3.3.3.1 Designating the Flash Update Area

To designate the Flash Update Area, enter the following console command:

```
console> set flash update area-number
```

where *area-number* is the number of the area on which you want the commands to operate.

If you want to preserve the contents of the Flash Boot Area until you have set up the new image and script files, set the Flash Update Area to use the other area of flash memory.

3.3.4 Loading from the Network

When you load a CMIP file from a network load host, the image is loaded into the Flash Update Area. If the Flash Update Area is not defined, the image is loaded into Area 1.

Note that the first time you boot the DECNIS the Flash Update Area is not defined, so the combined image is loaded into Area 1, which, by default, is the Flash Boot Area.

3.3.5 Example

In this example, you wish to load a new image and configuration script into flash memory, as the existing ones are out of date. However, you also wish to keep the existing image and script in flash memory, while you test the new ones.

The original image and scripts were loaded into Area 1, which is identified as the Flash Boot Area.

To update the DECNIS you should do the following:

1. Set Area 2 to be the Flash Update Area,
2. Use the UPDATE command to load the new image and scripts into Area 2.
3. When this area has been fully configured, set Area 2 to be the Flash Boot Area.

4. Restart the DECNIS.

The original image is still safe in Area 1, so if there is any problem with the new configuration you can set the Flash Boot Area back to Area 1 and continue to re-configure Area 2.

The following steps go through this procedure in more detail:

Step 1. Viewing the Initial Configuration

The first time the DECNIS is booted from flash memory the Flash Boot Area is Area 1 by default, and the Flash Update Area is not defined. The SHOW FLASH command display is:

```
console> show flash
Flash Boot Area       1
Flash Update Area     not setup

Status of 1st Flash Area
Image Index           0
Script Index          0

Flash Contents        Index   Name            Size       Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:29
  Flash free  850432 bytes
```

Step 2. Updating Flash With a New Image

Before you can use the Update command you must designate an area to be used as the Flash Update Area. Enter the following command:

```
console> set flash update 2
```

You can now do a network load or use the UPDATE command to load a new image or combined file:

```
console> update 16.36.0.138:/usr/users/flash/nis_nis116.sys
 About to overwrite flash area 2, Confirm ? y
Is it ok to clear the flash first ? y
Requested Flash Update
 Flash modify is in progress; it cannot be stopped. You can either observe
 the progress or type return to exit. The status can be checked using the
 Show Flash command and, if in progress, Update with no parameter returns
 to this point:

 Flash Update : erasing flash.
 Flash Update : waiting for file transfer
 Flash Update : transferring and writing to flash............
 Flash Update : skipping the MPC-I image.........
 Flash Update : transferring and writing to flash.
 Flash Update : operation completed ok
```

Step 3. Viewing Updated Flash Memory

When the update has completed you can see that the new files have been loaded into the Flash Update Area by using the SHOW FLASH command:

```
console> show flash
Flash Boot Area       1
Flash Update Area     2
Flash Status : operation completed ok

Status of 1st Flash Area
Image Index           0
Script Index          0

Flash Contents        Index   Name            Size       Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:29
  Flash free  850432 bytes

Status of 2nd Flash Area
Image Index           0
Script Index          0

Flash Contents        Index   Name            Size       Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:29
  sumchk ok  32415    2       script          7416      1996-08-27-21:42
  sumchk ok  04497    3       test_script     1024      1996-07-09-12:42
  Flash free  841216 bytes
```

Step 4. Adding Profiles and Scripts

When the new image has been copied you can use the ADD command to fetch additional profiles and script files into the Flash Update Area:

```
console> add mcnm_prf node.reo.dec.com:/usr/users/flash/mcnm_prf
Requested Flash Update
 Flash modify is in progress; it cannot be stopped. You can either observe
 the progress or type return to exit. The status can be checked using the
 Show Flash command and, if in progress, Update with no parameter returns
 to this point:

 Flash Update : waiting for file transfer
 Flash Update : operation completed ok
console>
```

You can use the SHOW FLASH command to verify that the new file has been added:

```
console> show flash
Flash Boot Area       1
Flash Update Area     2
Flash Status : operation completed ok

Status of 1st Flash Area
Image Index           0
Script Index          0
```

```
Flash Contents        Index   Name            Size       Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:29
  Flash free  850432 bytes

Status of 2nd Flash Area
Image Index           0
Script Index          0

Flash Contents        Index   Name            Size       Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:29
  sumchk ok  32415    2       script          7416      1996-08-27-21:42
  sumchk ok  04497    3       test_script     1024      1996-07-09-12:42
  sumchk ok  24778    4       mcnm_prf        1028      1582-10-15-00:28
  Flash free  839680 bytes
console>
```

Step 5. Using the Updated Image

When the Flash Update Area has been configured with the required scripts and profiles you can set the DECNIS to boot from this flash area using this console command:

```
console> set flash boot 2
console>
```

You can now restart the DECNIS using the new configuration. For example:

```
console> load -F
Confirm, reload the system ? y

%MOPFW, attempting program load from FLASH RAM...

%MOPFW, l o a d i n g from FLASH RAM...

%MOPFW, load completed
 time           : 0 seconds
 total size     : 41 blocks
 transfer       : 0x800


%FLASH, FLASH BOOT Program
%FLASH, decompressing image from flash

%FLASH, using flash area 2
 ....................
```

If there is no valid image in the Flash Boot Area, the DECNIS will load an image from the network and save it in the current Flash Boot Area.

Step 6. Reviewing Flash Memory

After the system has reloaded, the Flash Boot Area will be set to 2, but the Flash Update Area will not be defined. In this example the first flash area still contains the original image, since it was never cleared, and the second flash area contains the new configuration which was used to reload the system.

```
console> show flash
Flash Boot Area       2
Flash Update Area     not setup

Status of 1st Flash Area
Image Index           0
Script Index          0

Flash Contents        Index   Name            Size      Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:29
  Flash free  850432 bytes

Status of 2nd Flash Area
Image Index           0
Script Index          0

Flash Contents        Index   Name            Size      Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:29
  sumchk ok  32415    2       script          7416      1996-08-27-21:42
  sumchk ok  04497    3       test_script     1024      1996-07-09-12:42
  sumchk ok  24778    4       mcnm_prf        1028      1582-10-15-00:28
  Flash free  839680 bytes
```

Step 7. Deleting An Index Entry From Flash Memory

We want to delete the dummy entry `test_script` from the index. Since the Flash Update Area has not been defined, the DELETE command will not work yet.

```
console> delete 3
Flash Update Area must be set first
```

To delete an entry, you must enter the following commands:

```
console> set flash update 2
console> delete 3
 File index 3 in flash area 2 will be marked as deleted,
 but the space cannot be recovered, Confirm ? y
Requested Flash Update

 Flash Update : operation completed ok
```

The SHOW FLASH command reveals that Flash Update Area is Area 2, and that the entry has been deleted from the index in the 2nd Flash Area.

Remember that the memory has not been recovered.

```
console> show flash
Flash Boot Area       2
Flash Update Area     2
Flash Status : operation completed ok

Status of 1st Flash Area
Image Index           0
Script Index          0

Flash Contents        Index   Name            Size       Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:29
  Flash free  850432 bytes

Status of 2nd Flash Area
Image Index           0
Script Index          0

Flash Contents        Index   Name            Size       Date
  sumchk ok  02133    1       nis_v4.0-1B     3322310   1996-08-27-15:29
  sumchk ok  32415    2       script          7416      1996-08-27-21:42
  sumchk ok  24778    4       mcnm_prf        1028      1582-10-15-00:28
  Flash free  839680 bytes
console>
```

Step 8. Restoring The Original Image

If you want to return to using the original image, there is no need to load it from the load host, since it is still in flash memory. Simply set the Flash Boot Area to be the original area (Area 1).

```
console> set flash boot 1
```

4 Configuring SNMP on the DECNIS

4.1 Introduction

Any node in the same IP network as the DECNIS can use SNMP (Simple Network Management Protocol) to manage the DECNIS, provided the node complies with the SNMP standard as specified in RFC 1157. A node being used in this way is called a **network management station**.

The network management station can send SNMP Set requests to alter the configuration of the DECNIS, and Get requests to monitor its status.

4.2 MIBs

4.2.1 MIB Definition

An SNMP-compliant network device such as the DECNIS supports one or more Management Information Bases (MIBs). Each MIB is a collection of **objects**.

The values of these objects determine the behavior of corresponding aspects of the network device. The values can be read or altered by the network management station.

Example: The Bridge MIB is a collection of objects relevant to the bridging function of a network device. A network device which does not perform bridging would not need to support the Bridge MIB.

4.2.2 Groups and Objects

The objects in a MIB are divided into groups. Each group corresponds to a particular aspect of the network device.

Example: MIB-II has a System group, which contains objects relating to the network device as a whole, such as its physical location, the functions it performs, and so on.

A network device does not have to support all groups in a MIB. Generally, the MIB specifies which of its groups are mandatory.

4.3 Overview

Figure 4–1 shows a network management station being used to manage a DECNIS across an IP network.

Network Management Station

This is the system (host) from which network management instructions are sent to the DECNIS.

Network Management Software

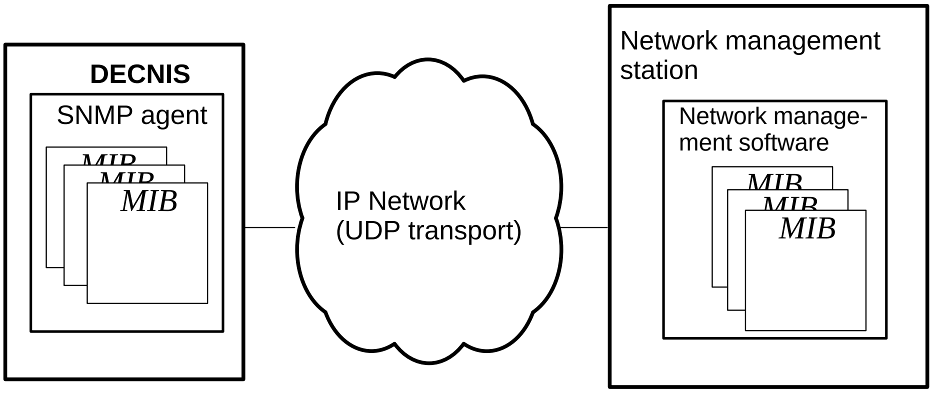
This is the SNMP-based management application running on the network management station.

Note that the MIBs supported by the DECNIS can be compiled into the management application: see the documentation for the management application for details of how to do this.

SNMP Agent

Any network device that has been configured to be manageable using SNMP must have an SNMP agent residing on it. In this case, the SNMP agent resides on the DECNIS.

Figure 4–1 Managing the DECNIS Using SNMP



CBN–0069–93–I

4.4 Access Control

4.4.1 Community Names

Each SNMP request sent to a network device contains a community name. Each network device is preconfigured with one or more community names, and the sort of access allowed for each (read-only, read-write, or no access).

If a network device receives an SNMP request containing an unknown community name, or a community name that is associated with the wrong sort of access, the network device will not respond.

4.4.2 Example

Network managers have configured a DECNIS with two community names, PUBLIC and PRIVATE. They have associated SNMP read-only access with PUBLIC, and read-write access with PRIVATE.

If the DECNIS receives an SNMP **read** request that contains the community name PRIVATE or PUBLIC, the request succeeds. If the DECNIS receives an SNMP read request that contains any other community name, the request fails.

If the DECNIS receives an SNMP **write** request that contains the community name PRIVATE, the request succeeds. If the DECNIS receives an SNMP write request that contains the community name PUBLIC, or any other community name, the request fails.

4.5 MIBs Supported by the DECNIS

4.5.1 MIBs and MIB Groups

The DECNIS supports the following MIBs and MIB groups:

- MIB-II (RFC 1213)

  MIB groups that the DECNIS supports in this MIB are:

  - system
  - interfaces
  - at
  - ip
  - icmp
  - udp
  - egp

  - transmission
  - snmp
- Bridge MIB (RFC 1493)

  MIB groups that the DECNIS supports in this MIB are:

  - dot1dBase
  - dot1dStp
  - dot1dTp
  - dot1dStatic
- FDDI MIB (RFC 1285 - draft)

  MIB groups that the DECNIS supports in this MIB are:

  - snmpFddiSMT
  - snmpFddiMAC
  - snmpFddiPORT
  - snmpFddiATTACHMENT
- DS3 MIB (RFC 1407)

  MIB groups that the DECNIS supports in this MIB are:

  - dsx3ConfigTable
  - dsx3CurrentTable
  - dsx3IntervalTable
  - dsx3TotalTable

  Note: The DECNIS provides Read-Only access to the dsx3ConfigTable
- DEC Vendor MIB elanext V2.7

  MIB groups that the DECNIS supports in this MIB are:

  - efddi

    efddiSMT
    efddiMAC
    efddiPORT
    efddiFDX
  - ebridge
  - eauth

4.5.2 Location of MIBs

The DEC Vendor MIB elanext V2.7 is supplied with the DECNIS kit. Table 4–1 gives the file name and location for each type of host on which the kit can be installed.

Table 4–1 Location of DEC Vendor MIB

Host	File Specification
OpenVMS	SYS$HELP:DEC_ELAN_MIB.V27_TXT
Digital UNIX	/usr/lib/dnet/dec_elan_mib.v27_txt
MS–DOS PCs and Windows 95/NT PCs	C:\ *install-dir* \ DOCS \ ELANMIB.TXT

where *install-dir* is the directory where the PC kit is installed.

You can access MIB-II, Bridge MIB, and FDDI MIB through ftp from the repository at nic.ddn.mil in the rfc directory. Check the rfc-index file for details of the MIBs available.

4.6 Traps

4.6.1 Definition

SNMP traps are messages that are sent by a network device when one of a prescribed list of changes in its operation occurs. The traps can be received by one or more systems in the same IP network as the originating device.

4.6.2 Traps Supported by the DECNIS

The DECNIS sends the traps specified in Table 4–2. Note that all traps include the IP address of the DECNIS.

Table 4–2 Traps Sent by the DECNIS

Trap	Meaning
coldStart	The DECNIS has rebooted.
warmStart	The SNMP entity of the DECNIS has been enabled.
linkUp	A data link has changed to state "up". The trap includes the SNMP MIB-II index number of the interface.

(continued on next page)

Table 4–2 (Cont.) Traps Sent by the DECNIS

Trap	Meaning
linkDown	A data link has changed to state "down". The trap includes the SNMP MIB-II index number of the interface.
authenticationFailure	The DECNIS has received an SNMP request that specifies an incorrect community name.
egpNeighborLoss	The DECNIS has marked an EGP peer relationship down. The trap includes the IP address of the EGP neighbor.

4.7 Configuring the DECNIS to Be Manageable Using SNMP

4.7.1 Introduction

This section describes the NCL commands required to allow the DECNIS to respond to SNMP requests from a management station.

It also describes how to use NCL and SNMP to configure the DECNIS to send traps.

You can issue these commands on a running system. You can also use the DECNIS configurator (text-based or Windows) to set up SNMP management. In this case, you must reboot the DECNIS before the changes will take effect.

4.7.2 Configuring the DECNIS to Respond to SNMP Requests

Follow these steps to configure the DECNIS to respond to SNMP requests from a remote management station:

1. Enable the SNMP entity:

   ```
   NCL> ENABLE NODE decnis SNMP
   ```

   (The SNMP entity is created automatically when the DECNIS is booted.)

2. Create COMMUNITY entities to control SNMP access to the DECNIS.

   The name of each COMMUNITY entity forms (part of) the community name. Each COMMUNITY entity specifies the type of SNMP access allowed to management stations sending that community name.

   Create COMMUNITY entities as follows:

   – Create the entity:

     ```
     NCL> CREATE NODE decnis SNMP COMMUNITY name
     ```

name is the name you use to refer to this entity. It also forms the first part of the community name that the DECNIS will try to validate when it receives an SNMP request.

Note that this name is case-sensitive; also, although nonalphanumeric characters are permitted in this name, Digital recommends that you use only alphanumeric characters. (Note also that you cannot use a wildcard (*) in an NCL command to show all COMMUNITY entities on the DECNIS, since * is a valid name for a COMMUNITY entity.)

– If you wish, set a second (nonreadable) part of the community name for this COMMUNITY entity by entering the following command:

```
NCL> SET NODE decnis SNMP COMMUNITY name -
_NCL> PRIVATE NAME private-name
```

private-name, when appended to the name of the COMMUNITY entity, forms the community name that the DECNIS will try to validate when it receives an SNMP request. Note that the PRIVATE NAME characteristic is write-only, and can contain nonalphanumeric characters. It is case-sensitive.

If you do not set this characteristic, it remains at its default (null).

– Specify the type of access for this community:

```
NCL> SET NODE decnis SNMP COMMUNITY name ACCESS access
```

access is READ-ONLY or READ-WRITE.

3. Enable each COMMUNITY entity:

```
NCL> ENABLE NODE decnis SNMP COMMUNITY name
```

4.7.3 Example

When the DECNIS receives an SNMP request, it checks the community name specified in the request, and permits or denies access accordingly.

You want to configure the DECNIS with two community names: one for read-only access, and one for read-write access.

1. Enable the SNMP entity:

```
NCL> ENABLE NODE decnis SNMP
```

2. Create a COMMUNITY entity for read-only access, called PUBLIC:

```
NCL> CREATE NODE decnis SNMP COMMUNITY public
```

3. Set the access for this community to read-only:

   ```
   NCL> SET NODE decnis SNMP COMMUNITY public -
   _NCL> ACCESS read-only
   ```

4. Enable the entity:

   ```
   NCL> ENABLE NODE decnis SNMP COMMUNITY public
   ```

5. Now create a community entity for read-write access, called PRIVATE:

   ```
   NCL> CREATE NODE decnis SNMP COMMUNITY private
   ```

6. For security, set a second, nonreadable part of the community name:

   ```
   NCL> SET NODE decnis SNMP COMMUNITY private-
   _NCL> PRIVATE NAME _secret
   ```

7. Set the access for this community to read-write:

   ```
   NCL> SET NODE decnis SNMP COMMUNITY private -
   _NCL> ACCESS read-write
   ```

8. Enable the entity:

   ```
   NCL> ENABLE NODE decnis SNMP COMMUNITY private
   ```

4.7.3.1 Result

If the DECNIS receives an SNMP read request containing the community name PUBLIC or PRIVATE_SECRET, access is allowed. A read request bearing any other community name will not succeed.

If the DECNIS receives an SNMP write request containing the community name PRIVATE_SECRET, access is allowed. A write request bearing any other community name will not succeed.

4.7.4 Using NCL to Configure the DECNIS to Send Traps

Note that use of NCL and SNMP are equivalent, and affect the same internal DECNIS database. If you set up trap destinations using NCL, you can use SNMP to show these destinations.

Follow these steps to configure the DECNIS to send traps:

1. Ensure that the SNMP entity is enabled:

   ```
   NCL> ENABLE NODE decnis SNMP
   ```

   (The SNMP entity is created automatically when the DECNIS is booted.)

2. Specify the IP addresses of the systems to which the DECNIS will send traps, together with the community name to be included in traps:

   ```
   NCL> SET NODE decnis SNMP TRAP SINKS -
   _NCL> {(DESTINATION= a.b.c.d, COMMUNITY = name), ...}
   ```

   a.b.c.d is the IP address of the system to which the DECNIS will send traps, and *name* is the community name to be included in all traps sent by the DECNIS. Note that this community name will be the same irrespective of the trap destination. If you enter different community names in the above command, only the last one you enter will be included.

   You may enter up to four IP address/community name combinations, but all community names except the last will be ignored.

   If you do not wish traps from the DECNIS to include a community name, specify a null community name in the final IP address:

   ```
   NCL> SET NODE decnis SNMP TRAP SINKS -
   _NCL> {..., (DESTINATION= a.b.c.d, COMMUNITY = "")}
   ```

   You can use the ADD NCL command to add an IP address/community name entry to an existing set; the REMOVE command removes an IP address/community name entry from an existing set.

3. If you wish the DECNIS to send authenticationFailure traps, you must issue the following command:

   ```
   NCL> SET NODE decnis SNMP -
   _NCL> GENERATE AUTHENTICATION FAILURE EVENTS true
   ```

   By default, the DECNIS does not send authenticationFailure traps.

4.7.5 Using SNMP to Configure the DECNIS to Send Traps

Note that use of NCL and SNMP are equivalent, and affect the same internal DECNIS database. If you set up trap destinations using SNMP, you can use NCL to show these destinations.

Table 4–3 shows the MIB variables involved in configuring the DECNIS to send traps. How you read and set these variables depends on the management application software running on the network management station.

Table 4–3 SNMP Traps: MIB Variables

MIB Group	MIB Table	MIB Variable	Comments
		DEC Vendor MIB elanext V2.7	
eauth	(scalar)	eauthTrapCommunity	Set the value of TrapCommunity_0 to the community name to be included in the traps sent by the DECNIS.
eauth	TrapUserTable	eauthTrapUserAddr	Create a TrapUserAddr_ *a.b.c.d*, with a value of *a.b.c.d*, where *a.b.c.d* is the IP address to which traps are to be sent.
		eauthTrapUserStatus	Set this to "invalid" to delete the relevant trap destination.
		MIB-II	
SNMP	(scalar)	snmpEnableAuthenTraps	Set this to "enabled" if you want the DECNIS to generate authentication-failure traps.

4.8 Using SNMP to Manage the DECNIS

Section 8.27 describes the use of SNMP to manage data links on the DECNIS.

Section 10.26 describes the use of SNMP to manage the IP routing functions of the DECNIS.

Section 21.17 describes the use of SNMP to manage the bridge functions of the DECNIS.

Note

The above sections do not describe in detail how to use SNMP. They assume that you are familiar with and have some experience of using SNMP.

For more information, refer to the relevant RFCs, and the documentation for the network management software you are using.

4.9 clearVISN

clearVISN is Digital's policy-based element manager. It consists of a suite of SNMP-based network management applications for managing devices within a network.

Because of its extensive SNMP and MIB support, the DECNIS can be managed using the clearVISN Router Manager. The clearVISN Router Manager is an SNMP router management application with multivendor support for routers. Its features include:

- **Device discovery**. When given an IP address, clearVISN discovers the appropriate IP device.
- **Path Tracing**. You can trace paths between IP devices in the network, to help find problems in network links.
- **Performance Monitoring**. clearVISN uses MIB information to determine bandwidth utilization on a per interface or per protocol basis, and automatically create a utilization graph.
- **Alarms and Reports**. You can set thresholds for automatic alarms and to show network events, for example, percentage line utilization thresholds. You can also generate periodic reports according to predetermined criteria.
- **Configuration Manager**. This provides management of multiple device configuration files, including DECNIS configuration and image files.
- **Telnet capability**. This allows you to connect to the DECNIS console and the NCL command line interface to manage running DECNIS systems.
- **Policy-based management**. This allows procedures to be set up to simplify the management of complex network devices and technologies.

Refer to the clearVISN documentation for more information.

5

Setting Up Event Logging on the DECNIS

5.1 Event Logging on the DECNIS

This chapter describes how to set up event logging on the DECNIS.

A list of the event messages sent by the DECNIS is supplied on line. The locations of the event message files are as follows:

Load Host	Location
MS–DOS, Windows 95 and Windows NT PCs	*install-directory*\ DOCS\ EVMSG31A.TXT *install-directory*\ DOCS\ EVMSG31B.TXT *install-directory*\ DOCS\ EVMSG31C.TXT
OpenVMS	SYS$HELP:NIS$EVENTS.TXT
Digital UNIX	/usr/lib/dnet/nis_event.txt

5.1.1 Outbound Event Streams

Outbound event streams allow events to be passed from the DECNIS to an event sink.

5.1.2 Example

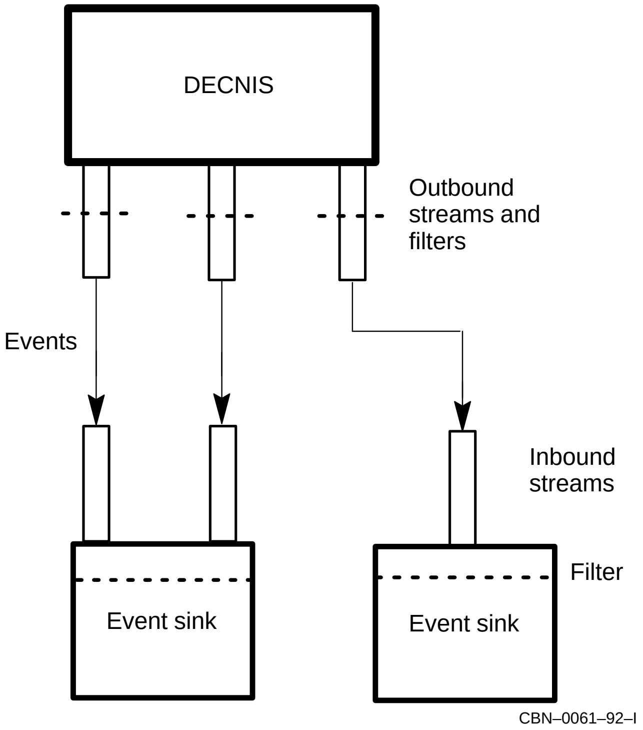
Figure 5–1 illustrates a DECNIS on which three event streams have been set up, to allow events to be passed to two event sinks.

5.2 Event Sinks

5.2.1 Requirement to Set Up Event Sinks

You must set up the event sink on the sink node before you can use event logging. Appendix C contains the NCL commands required to set up event sinks. For more information, refer to the documentation for your sink node.

Figure 5–1 Event Streams on the DECNIS



5.2.2 Structure of Tower Set for an Event Sink

If the node name of the event sink has not been entered in the Known Towers database (see Section 1.11) of the DECNIS, then you must specify the event sink as a tower set.

A tower set consists of one or more towers. An event sink tower is:

```
([DNA_CMIP-MEN], [DNA_SESSIONCONTROLVn, NUMBER = 82],
[DNA_NSP], [DNA_OSINETWORK, NSAP-address])
```

n	is 3 if the event sink is a DECnet/OSI node. is 2 if the event sink is a Phase IV node.
NSAP-address	is the NSAP address of the event sink.

5.2.3 Example

Suppose the (DECnet/OSI) node that is to act as the event sink for the DECNIS has the following two NSAP addresses:

37:12345:02-00:AA-02-14-78-66-11:20
37:12345:02-00:AA-02-14-78-66-10:20

The tower set representing this event sink consists of two towers, as follows:

```
SET NODE decnis EVENT DISPATCHER -
OUTBOUND STREAM stream-name SINK ADDRESS -
{([DNA_CMIP-MEN], [DNA_SESSIONCONTROLV3, NUMBER = 82], -
[DNA_NSP], -
[DNA_OSINETWORK, 37:12345:02-00:AA-02-14-78-66-11:20]), -
([DNA_CMIP-MEN], [DNA_SESSIONCONTROLV3, NUMBER = 82], -
[DNA_NSP], [DNA_OSINETWORK, 37:12345:02-00:AA-02-14-78-66-10:20]) -
}
```

where *decnis* is the name of the DECNIS.

5.2.4 Using DECmcc as the Event Sink

To set up DECmcc as the event sink for the DECNIS, create the tower as above, but specify NAME=MCC_EVL_SINK instead of NUMBER=82.

Example

Suppose the (DECnet/OSI) node running DECmcc that is to act as the event sink for the DECNIS has the following NSAP address:

37:12345:02-00:AA-02-14-78-66-09:20

The tower set representing this event sink consists of the following tower:

```
SET NODE decnis EVENT DISPATCHER -
OUTBOUND STREAM stream-name SINK ADDRESS -
{([DNA_CMIP-MEN], [DNA_SESSIONCONTROLV3, NAME=MCC_EVL_SINK], -
[DNA_NSP], -
[DNA_OSINETWORK, 37:12345:02-00:AA-02-14-78-66-09:20])}
```

where *decnis* is the name of the DECNIS.

5.3 Setting Up Event Logging

5.3.1 Introduction

This section describes how to set up an outbound event stream on the DECNIS.

5.3.2 Procedure

Follow these steps to set up event logging on the DECNIS.

1. Ensure that the EVENT DISPATCHER entity exists and is enabled:

   ```
   NCL> CREATE NODE decnis EVENT DISPATCHER
   NCL> ENABLE NODE decnis EVENT DISPATCHER
   ```

   where *decnis* is the name of the DECNIS.

2. Create the outbound event stream:

   ```
   NCL> CREATE NODE decnis EVENT DISPATCHER -
   _NCL> OUTBOUND STREAM stream-name
   ```

   where *stream-name* is the name of the outbound event stream.

3. Specify the sink to be used to log the events from this event stream.

   You can specify the sink as a node name, or as a tower set.

   #### As a Node Name

   To specify the sink as a node name, issue the following command:

   ```
   NCL> SET NODE decnis EVENT DISPATCHER -
   _NCL> OUTBOUND STREAM stream-name SINK NODE sink-node
   ```

   where *sink-node* is the node name of the sink node as defined in the Known Towers database (see Section 1.11).

   #### As a Tower Set

   To specify the sink as a tower set, issue the following command:

   ```
   NCL> SET NODE decnis EVENT DISPATCHER -
   _NCL> OUTBOUND STREAM stream-name SINK ADDRESS {(tower1), (tower2), ...}
   ```

   where *tower1* and *tower2* are towers that describe the protocols and NSAP addresses used to communicate with the event sink.

4. Enable the OUTBOUND STREAM entity:

   ```
   NCL> ENABLE NODE decnis EVENT DISPATCHER -
   _NCL> OUTBOUND STREAM stream-name
   ```

5.4 Disconnecting the DECNIS from the Event Sink

5.4.1 Introduction

This section describes how to break the connection between the outbound event stream and the event sink.

5.4.2 Breaking the Connection Immediately

Issue the following command to destroy immediately the connection between an outbound event stream and its event sink:

```
NCL> DISCONNECT NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name
```

This will cause events in transit to be lost. You should issue this command if you have problems with your sink node and wish to specify a new sink node.

5.4.3 Breaking the Connection in an Orderly Manner

Issue the following command to achieve an orderly shutdown of the connection between an outbound event stream and its event sink:

```
NCL> SHUTDOWN NODE decnis EVENT DISPATCHER OUTBOUND STREAM stream-name
```

This breaks the connection for the outbound stream once all events in transit have been received. However, if the connection is faulty, the command may take a long time to execute.

5.4.4 Reestablishing the Connection

The DECNIS will attempt to reestablish the connection automatically when the CONNECT RETRY TIMER expires (see Section 5.6).

You can manually reestablish the connection by issuing the following command:

```
NCL> CONNECT NODE decnis EVENT DISPATCHER OUTBOUND STREAM stream-name
```

5.5 Disabling Event Streams

5.5.1 Introduction

Disable an event stream to prevent any events from being sent from the event stream to the event sink. The DECNIS can still send events to an event sink by using another event stream.

5.5.2 Procedure

Issue the following command to disable an event stream:

```
NCL> DISABLE NODE decnis EVENT DISPATCHER -
_NCL> OUTBOUND STREAM stream-name METHOD = method
```

where *method* is one of:

- ABORT

  The connection is destroyed at once, and all events in transit are lost. This is similar to using the DISCONNECT command (see Section 5.4.2).

- ORDERLY

  This is the default method of disabling an event stream. It allows all events in transit to be received by the sink before the connection is broken (see the SHUTDOWN command in Section 5.4.3). However, this directive may take a very long time to execute, particularly if the event sink is faulty.

To prevent immediately the DECNIS from sending any events to any event sink, use the following command:

```
NCL> DISABLE NODE decnis EVENT DISPATCHER -
_NCL> OUTBOUND STREAM * METHOD = ABORT
```

5.6 Connection Timers

5.6.1 Introduction

The connection between the event source and the sink is controlled by two timers, whose values can be changed as required.

5.6.2 Connect Retry Timer

This controls how often the outbound stream attempts to make connections to the sink.

Note

The CONNECT TIMER ENABLED characteristic of the OUTBOUND STREAM entity must be set to TRUE for this timer to operate. If it is set to FALSE, the outbound event stream will not make connection retries.

5.6.2.1 Changing the Connect Retry Timer

Set the connect retry timer as follows:

```
NCL> SET NODE decnis EVENT DISPATCHER OUTBOUND STREAM stream-name -
_NCL> CONNECT RETRY TIMER n
```

where:

decnis is the name of the DECNIS.

n is a decimal number between 1 and 65,535, and specifies the number of seconds to wait between attempts to create connections. The default is 120 seconds.

5.6.3 Disconnect Timer

This controls the length of time to wait before disconnecting idle connections.

5.6.3.1 Setting the Disconnect Timer

Set the disconnect timer as follows:

```
NCL> SET NODE decnis EVENT DISPATCHER OUTBOUND STREAM stream-name -
_NCL> DISCONNECT TIMER n
```

where:

decnis is the name of the DECNIS.

n is a decimal number greater than or equal to 1. If no events are logged during this number of seconds, the connection is dropped. Disable this timer by using a value of 0; the connection will then not be dropped automatically.

5.7 Event Filtering on the DECNIS

5.7.1 Introduction

You can set up specific, global, and catchall filters to determine which events generated by the DECNIS get sent to the event sink.

5.7.2 Specific Filters

Specific filters specify the following:

- A specific named entity (for example, a specific routing circuit on the DECNIS).
- An event that can be generated by that entity.
- Whether the event is to be passed, blocked or ignored.

5.7.3 Global Filters

Global filters specify the following:

- An entity class (for example, all routing circuits on the DECNIS).
- An event that can be generated by that class of entity.
- Whether the event is to be passed, blocked or ignored.

5.7.4 Catchall Filter

A catchall filter specifies whether the events not specified in a specific or global filter should be passed or blocked.

5.7.5 Operation of Event Filters

Figure 5–2 illustrates how event filters operate on the DECNIS.

5.7.6 Recommendation

You can also set up filters on event sinks. However, Digital recommends that you set up event sinks so that all events are received, and all filtering is done on the outbound stream. This reduces unnecessary network traffic and wasted resources for events that will be discarded at the sink.

5.8 Events Blocked by Default

5.8.1 List of Blocked Events

By default, global filters are set to block the events shown in Table 5–1 on all DECNIS outbound streams.

Figure 5–2 Operation of Event Filters in the Outbound Stream

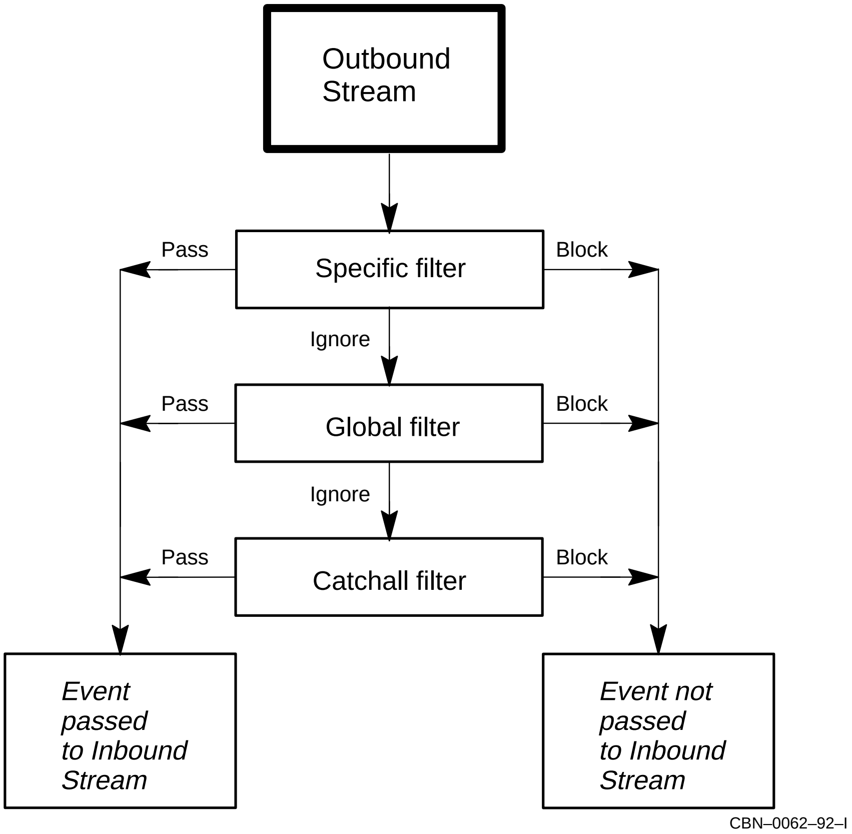


Table 5–1 DECNIS Events Blocked by Default

Entity Class	Event Name
ROUTING	Address Unreachable PDU Discard
	Aged PDU Discard
	IP Address Unreachable Packet Discard
	IP Source Address Error Packet Discard
	PhaseIV Translation Failure
ROUTING CIRCUIT	Adjacency State Change
	ID Reachability Change
	Initialization Failure
CSMA-CD STATION	Unrecognized Multicast Destination PDU
FDDI STATION	Unrecognized Multicast PDU Destination
NSP LOCAL NSAP REMOTE NSAP	Remote Protocol Error

5.8.2 Overriding Default Blocking

If you want to pass some or all of these events, insert the appropriate NCL commands in the NIS_*client-name*_EXTRA_SET.NCL file.

You can override the blocks entirely by setting up global filters, as described in Section 5.10.4, or you can override the blocks for specific instances of the entity, as described in Section 5.9.4.

5.9 Setting Up Specific Event Filters

5.9.1 Introduction

This section describes how to set up specific filters for events. A specific filter passes or blocks events from a particular entity instance, for example, a particular routing circuit.

5.9.2 To Block Events

Issue this command to set up a specific filter to block events:

```
NCL> BLOCK NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name SPECIFIC FILTER = -
_NCL> ((entity-instance), event)
```

where:

decnis	is the name of the DECNIS.
entity-instance	is the name of the entity from which events are to be blocked.
event	is the name of the event to be blocked.

5.9.3 Example: Blocking

To prevent the Circuit Change event on routing circuit Pepper from being sent to the sink, irrespective of any global or catchall filters, issue the following command:

```
NCL> BLOCK NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name SPECIFIC FILTER = -
_NCL> ((NODE decnis ROUTING CIRCUIT pepper), circuit change)
```

5.9.4 To Pass Events

Issue this command to set up a specific filter to pass events:

```
NCL> PASS NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name SPECIFIC FILTER = -
_NCL> ((entity-instance), event)
```

where:

decnis	is the name of the DECNIS.
entity-instance	is the name of the entity from which events are to be passed.
event	is the name of the event to be passed.

5.9.5 Example: Passing

To allow the Circuit Change event on routing circuit Pepper to be sent to the sink, irrespective of any global or catchall filters, issue the following command:

```
NCL> PASS NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name SPECIFIC FILTER = -
_NCL> ((NODE decnis ROUTING CIRCUIT pepper), circuit change)
```

5.10 Setting Up Global Event Filters

5.10.1 Introduction

This section describes how to set up global filters for events. A global filter passes or blocks events from all instances of a particular entity class, for example, all routing circuits.

5.10.2 To Block Events

Issue this command to set up a global filter to block events:

```
NCL> BLOCK NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name GLOBAL FILTER = -
_NCL> ((entity-class), event)
```

where:

decnis	is the name of the DECNIS.
entity-class	is the name of the entity class from which events are to be blocked.
event	is the name of the event to be blocked.

5.10.3 Example: Blocking

To prevent the Circuit Change event on all routing circuits from being sent to the sink, irrespective of any catchall filter, issue the following command:

```
NCL> BLOCK NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name GLOBAL FILTER = -
_NCL> ((NODE, ROUTING, CIRCUIT), circuit change)
```

Note that Circuit Change events would still be logged for any routing circuits on which a specific filter had been set to pass Circuit Change events.

5.10.4 To Pass Events

Issue this command to set up a global filter to pass events:

```
NCL> PASS NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name GLOBAL FILTER = -
_NCL> ((entity-class), event)
```

where:

decnis	is the name of the DECNIS.
entity-class	is the name of the entity class from which events are to be passed.
event	is the name of the event to be passed.

5.10.5 Example: Passing

To allow the Circuit Change event on all routing circuits to be sent to the sink, irrespective of any catchall filter, issue the following command:

```
NCL> PASS NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name GLOBAL FILTER = -
_NCL> ((NODE, ROUTING, CIRCUIT), circuit change)
```

Note that Circuit Change events would not be logged for any routing circuits on which a specific filter had been set to block Circuit Change events.

5.11 Setting Up a Catchall Event Filter

5.11.1 Introduction

This section describes how to set up a catchall filter for events. A catchall filter passes or blocks all events from a particular outbound event stream.

5.11.2 To Block Events

Issue this command to set up a catchall filter to block all events:

```
NCL> SET NODE decnis EVENT DISPATCHER -
_NCL> OUTBOUND STREAM stream-name CATCH ALL FILTER BLOCK
```

where:

decnis is the name of the DECNIS.

stream-name is the name of the outbound event stream.

Note that events would still be logged if any specific or global filters had been set to pass them.

5.11.3 To Pass Events

Issue this command to set up a catchall filter to pass events:

```
NCL> SET NODE decnis EVENT DISPATCHER -
_NCL> OUTBOUND STREAM stream-name CATCH ALL FILTER PASS
```

where:

decnis is the name of the DECNIS.

stream-name is the name of the outbound event stream.

Note that events would still be blocked if any specific or global filters had been set to block them.

5.12 Removing Specific and Global Event Filters

5.12.1 Introduction

This section describes how to use the IGNORE command to remove specific or global event filters.

5.12.2 Removing Specific Event Filters

Issue the following command to remove a specific event filter:

```
NCL> IGNORE NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name SPECIFIC FILTER = -
_NCL> ((entity-instance), event)
```

where:

decnis is the name of the DECNIS.

entity-instance is the name of the entity specified in the specific filter.

event is the name of the event specified in the specific filter.

Example

If you have created a specific filter that blocks the Adjacency State Change event from routing circuit Pepper, use the following command to delete this specific filter:

```
NCL> IGNORE NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name SPECIFIC FILTER = -
_NCL> ((NODE decnis ROUTING CIRCUIT pepper), adjacency state change)
```

Events from routing circuit Pepper will now be passed or blocked according to any global filtering for routing circuits (see Section 5.7.3).

5.12.3 Removing Global Event Filters

Issue the following command to remove a global event filter:

```
NCL> IGNORE NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name GLOBAL FILTER = -
_NCL> ((entity-class), event)
```

where:

decnis is the name of the DECNIS.

entity-class is the name of the entity class specified in the global filter.

event is the name of the event specified in the global filter.

Example

If you have created a global filter that blocks the Adjacency State Change event from all routing circuits, use the following command to delete this global filter:

```
NCL> IGNORE NODE decnis EVENT DISPATCHER OUTBOUND STREAM -
_NCL> stream-name GLOBAL FILTER = -
_NCL> ((NODE, ROUTING, CIRCUIT), adjacency state change)
```

Events from routing circuit Pepper will now be passed or blocked according to the catchall filter for this event stream (see Section 5.7.4), unless there is an appropriate specific filter.

For example, if there is a specific filter that passes the Adjacency State Change event on routing circuit Pepper, this event will always be passed, irrespective of any global or catchall filtering.

5.13 Testing Event Streams and Filters

5.13.1 Introduction: TESTEVENT Command

Use the TESTEVENT command to check whether a particular event from a particular entity will be passed or blocked, and to see the filter type used to pass or block it.

5.13.2 Command Structure

The TESTEVENT command is as follows:

```
NCL> TESTEVENT NODE decnis EVENT DISPATCHER -
_NCL> OUTBOUND STREAM stream-name EVENT = -
_NCL> ((entity-instance), event)
```

where:

decnis	is the name of the DECNIS.
entity-instance	is the name of the entity to be tested.
event	is the name of the event to be tested.

5.13.3 Example

Issue the following command to check what happens to the Circuit Change event on routing circuit Pepper:

```
NCL> TESTEVENT NODE decnis EVENT DISPATCHER -
_NCL> OUTBOUND STREAM stream-name EVENT = -
_NCL> ((NODE decnis ROUTING CIRCUIT pepper), circuit change)
```

If this event is blocked by a specific filter, the command will return the following information:

```
Action = BLOCK
Type = SPECIFIC
```

6
System-Level Management

6.1 Introduction

This chapter describes those aspects of the DECNIS that relate to the system as a whole. These are:

- System-level security
- Assigning a name to the DECNIS
- Setting the system time on the DECNIS
- The system group of MIB-II

6.2 Security for NCL

6.2.1 Need for Security

Security for the use of NCL is required to prevent unauthorized changes to the DECNIS.

6.2.2 NCL Security Mechanism

In order to restrict the use of NCL commands that alter the system configuration, you need to create a user name and password on the management listener, OBJ_19. These are sometimes referred to as the network management user name and password.

Note that you cannot restrict the use of the NCL SHOW command.

6.3 Setting Up Security Within the Configurators

The DECNIS text-based configurator requires you to set up the network management user name and password in the Configuration Options section.

The clearVISN DECNIS configurator does not require you to set up a network management user name and password. If you wish, you can set up the user name and password on the Security tab page under the **System** button on the Main Navigation window.

6.3.1 Setting Up the User Name and Password using NCL Commands

To set up or change the user name and password using NCL commands, enter the following:

```
NCL> SET NODE decnis SESSION CONTROL APPLICATION OBJ_19 -
_NCL> USER NAME username, PASSWORD password
```

where: *decnis* is the name of the DECNIS and *username* and *password* are the user name and password that users must supply when they enter NCL commands on the DECNIS.

6.3.2 Result of Creating the NCL User Name and Password

To use NCL on the DECNIS, users will need to enter the user name and password in the usual way for their operating system. Refer to Section 1.7.1 for examples.

In addition, if you have not set up a console password, you must supply the network management password when you use Telnet to connect to the DECNIS console.

6.4 Security for the Common Trace Facility

6.4.1 Need for Security

Security for the Common Trace Facility (CTF) is required to prevent unauthorized users from monitoring the data passing through the DECNIS.

6.4.2 CTF Security Mechanism

Access to the Common Trace Facility (CTF) is restricted by creating a user name and password on the CTF listener, OBJ_54. You supply this user name and password when you use the DECNIS configurator (text-based or Windows) to configure the DECNIS.

6.4.3 Changing the User Name and Password for CTF

To set up or change the user name and password on the CTF listener, using NCL commands, enter the following:

```
NCL> SET NODE decnis SESSION CONTROL APPLICATION obj_54 -
_NCL> USER NAME username, PASSWORD password
```

where: *decnis* is the name of the DECNIS and *username* and *password* are the user name and password that users must quote when they use CTF on the DECNIS.

Result

To use CTF on the DECNIS, users will need to quote the user name and password in the usual way for their operating system.

6.4.4 Example

To start CTF on a node called ORG:.SOUTH.SALES, which has a user name SMITH and password SECRET, specifying a tracepoint "HDLC LINK THIS-LINK", enter:

```
$ TRACE START org:.south.sales"smith secret"::"HDLC LINK this-link"
```

6.5 Preventing Unauthorized Loading

6.5.1 Need for Security

By default, any user on an adjacent system can send a LOAD command to the DECNIS, optionally specifying a new system image.

You can set up a verification value (password) on the DECNIS. This will ensure that only those users who supply this verification will be permitted to issue a LOAD command to the DECNIS.

6.5.2 Setting a MOP Verification Value

Set up a MOP verification value as follows:

```
NCL> SET NODE decnis MOP VERIFICATION value
```

where *decnis* is the name of the DECNIS, and *value* is an octet string of 8 bytes. An example verification value is %X8877665544332211.

6.5.3 Do Not Include Commands in NCL Script File

The MOP verification value is stored in nonvolatile memory on the DECNIS, and does not change when the DECNIS is rebooted. You should not add this command to NIS_*mop-client-name*_EXTRA_SET.NCL.

Clearing Nonvolatile Memory

To clear the nonvolatile memory of the DECNIS, power up the hardware with the Dump button held in. See the *Installation and Service Manual*.

6.6 Assigning a Name to the DECNIS

By default, when the DECNIS first boots, it does not know its name. Any events logged from the DECNIS will appear as logged from node 0.

After the DECNIS is first booted, enter the following command to place the DECNIS name in its nonvolatile memory:

```
NCL> RENAME NODE phase-iv-name NEW NAME decnis
```

where *phase-iv-name* is the Phase IV name for the DECNIS, and *decnis* is its DECdns name (including the namespace name) or local namespace name.

6.7 Security for SNMP

6.7.1 Security for SNMP Requests

You can use any of the following to set up community names to validate SNMP Get and Set commands sent to the DECNIS:

- The DECNIS text-based configurator
- The clearVISN DECNIS configurator
- NCL commands

The DECNIS will not respond to SNMP requests that contain an incorrect community name.

Section 4.7 describes how to set up community names.

6.7.2 Security for Traps

You can use any of the following to set up a community name that will be sent in traps from the DECNIS:

- The DECNIS text-based configurator
- The clearVISN DECNIS configurator
- NCL commands
- SNMP requests

The management stations to which the DECNIS traps are sent may then use this community name to reject or accept the trap.

Section 4.7 describes how to set up a community name in traps.

6.7.3 Secure Connections

Secure Connections is a utility that enables you to set up filtering to permit or deny network connect requests to or between network domains. Secure Connections can filter connect requests for IP, DECnet, and OSI.

Secure Connections works by enabling you to:

- Combine nodes, circuits, applications, and users into groups that share the same security requirements.
- Define access rules that apply to each group.

Refer to the *DECNIS Introduction and Glossary* for a more detailed description of Secure Connections.

6.7.3.1 Setting Up Secure Connections

You can only set up Secure Connections access rules in the clearVISN DECNIS configurator. Follow these steps:

1. Run the clearVISN DECNIS configurator.
2. Click the **Secure Connections** button on the Main Navigation window.
3. Enter Secure Connections information.

Refer to the *clearVISN DECNIS Configurator User Guide* for more information.

6.8 Setting the System Time on the DECNIS

You can check the system time of the DECNIS by checking the output of NCL SHOW commands, checking the events that are logged, or by issuing the following command:

```
NCL> SHOW NODE decnis DTSS CURRENT TIME
```

where *decnis* is the name of the DECNIS.

If the system time of the DECNIS is incorrect, use the following command to set it to the correct value:

```
NCL> UPDATE NODE decnis DTSS TIME yyyy-mm-dd-hh:mm:ss+a:b
```

where *yyyy-mm-dd-hh:mm:ss* is the year, month, day, and local time in hours, minutes and seconds. You must specify all parts of the date and time.

The variable +*a:b* is the TDF (time differential factor), and is the number of hours:minutes by which UTC time (coordinated universal time) differs from local time. Use a + sign if local time is ahead of UTC; use a – sign if local time is behind UTC.

Example

To set the system time of the DECNIS to a local time of 10.30 am, August 17, 1994, when local time is 5 hours behind UTC time, issue the following command:

```
NCL> UPDATE NODE decnis DTSS TIME 1994-08-17-10:30:00-5:0
```

6.9 MIB-II: System Group

You can use SNMP to set the following objects in the system group of MIB-II:

MIB Variable	Comments
	MIB-II, system group
sysContact	Specifies the contact person for the DECNIS.
sysName	Specifies the name by which the DECNIS will be known. If you use DECdns, it is recommended that you use the DECdns node name.
sysLocation	Specifies the physical location of the DECNIS.

7

Packet Prioritization on the DECNIS

7.1 Introduction

This chapter describes how to set up the packet prioritization function of the DECNIS.

7.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

7.2 Definition

By prioritizing packets, you can control the order in which the DECNIS transmits the packets.

For example, if Telnet traffic is more important than file transfer traffic, you assign a high priority to Telnet and a low priority to file transfer. This means that the DECNIS will always transmit Telnet packets before file transfer packets.

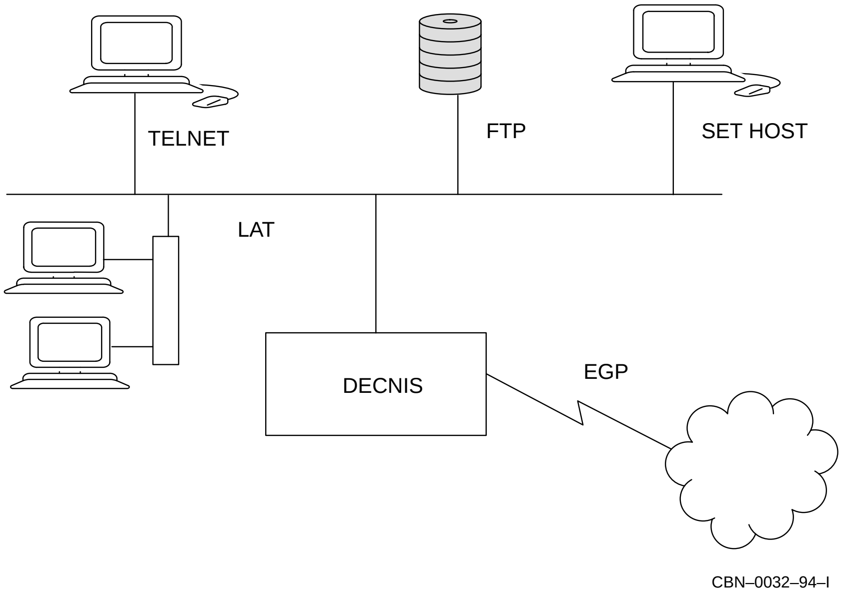
Note: You cannot set the priority of routing control or bridge control packets. These control packets always have a higher priority than any data packets.

Packet prioritization is supported on the following Network Interface Cards: WANcontroller 622, WANcontroller 614, WANcontroller 618, LANcontroller 601 and LANcontroller 602.

7.3 Why Use Packet Prioritization?

Figure 7–1 shows part of a typical network in which there are five different types of traffic.

Figure 7–1 Typical Network Traffic



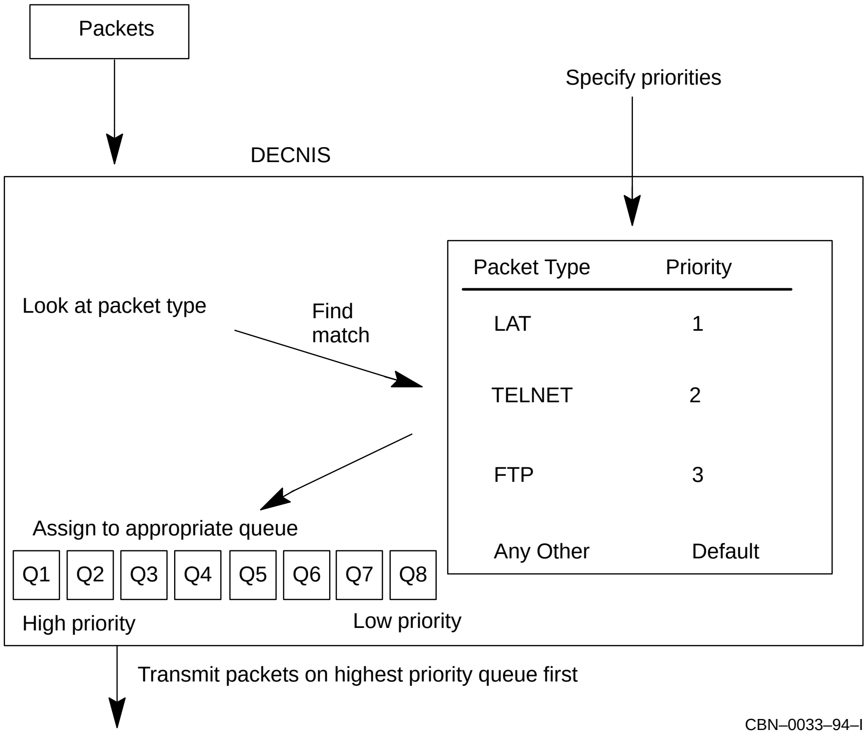
When there is a lot of different traffic on the network, some applications may experience delays. For example, terminal users may suffer delays to their screen display whilst a file transfer is taking place.

To reduce these delays to the screen, you can make terminal traffic a higher priority than file transfer traffic.

7.4 A Simple Overview of Packet Prioritization

Figure 7–2 gives a simple overview of how packet prioritization works.

Figure 7–2 Simple Overview of Prioritization



You specify the priority you want each type of packet to have. For example:

- LAT packets—Priority 1
- Telnet packets—Priority 2
- FTP packets—Priority 3

When the DECNIS receives a packet, it checks for a match with the types you specified and then assigns the relevant priority for that type. If it does not find a match, it uses the default priority.

The packets are assigned to a queue appropriate to the priority. High-priority packets are assigned to queue 1 and low-priority packets to queue 8. Packets on queue 1 are transmitted first and packets on queue 8 are transmitted last.

Section 7.5 describes in more detail how packet prioritization works.

7.5 How the PRIORITY Entities Control Packet Prioritization

Packet prioritization is controlled by the PRIORITY module and its subentities.

Figure 7–3 shows how the PRIORITY entities control the packets that the DECNIS receives.

1. Each PACKET entity contains a description of a packet type that is to be prioritized. When the DECNIS receives a packet, it looks for a PACKET entity with a description that matches the packet type.

   If the DECNIS finds a matching PACKET entity, it takes the "match action" associated with the PACKET entity. The "match action" associates the packet with a GROUP entity.

   If the DECNIS does not find a matching PACKET entity, it associates the packet with a DEFAULT CLASS characteristic.

   For example, in Figure 7–3, if the DECNIS received a Telnet packet, it would find the matching PACKET entity "telnet" and associate the packet with the GROUP entity "interactive". However, if the DECNIS received an ICMP packet, it would not find a matching PACKET entity and would associate the packet with an IP DEFAULT CLASS characteristic.

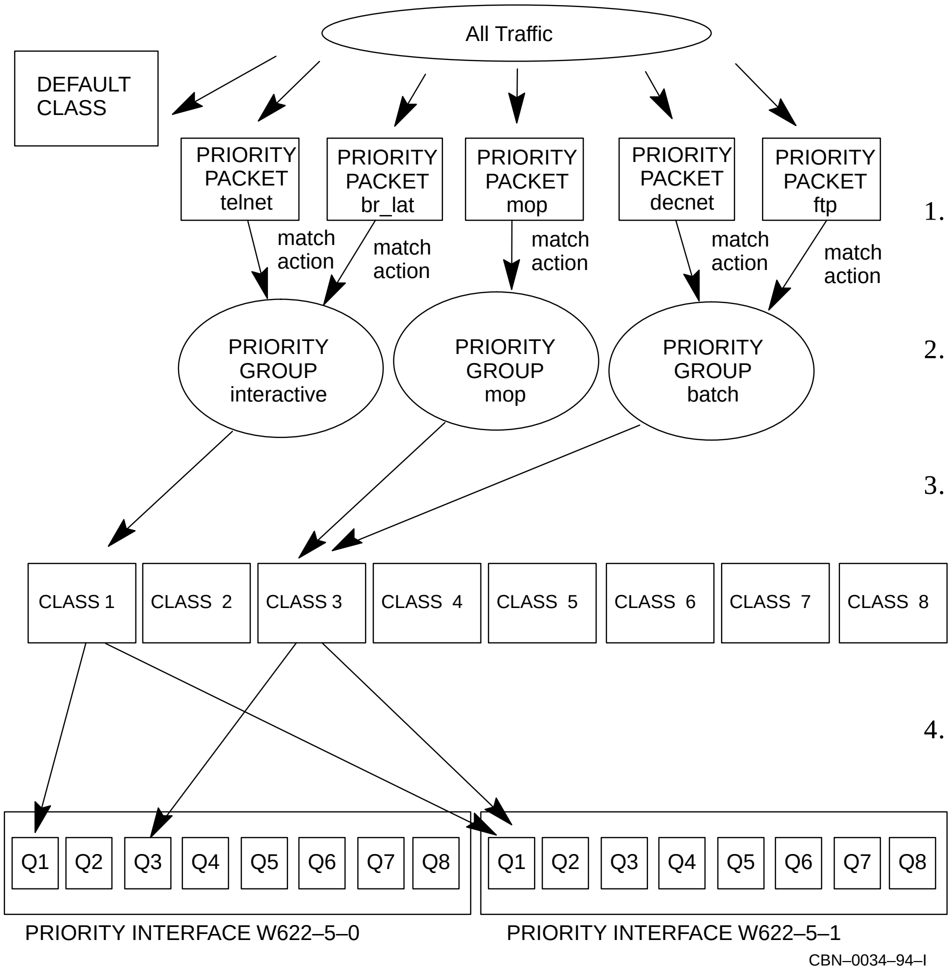
   Section 7.6 describes the different packet types that can be prioritized and the DEFAULT CLASS characteristics.

2. One or more PACKET entities can be associated with the same GROUP entity. This allows you to group packets of a similar type. For example, in Figure 7–3, "br_lat" and "telnet" are both associated with the same GROUP, "interactive".

3. Each GROUP entity is associated with a class. There are eight classes. Different GROUP entities can be associated with the same class, for example, in Figure 7–3, "mop" and "batch" are both associated with class 3.

4. Each class is associated with a queue on an INTERFACE entity. Each INTERFACE entity is associated with a communications port on a Network Interface Card, for example, W622-5-0.

Figure 7–3 How Prioritization Works



Each interface has eight queues. A class can be associated with any queue. By default, class 1 is associated with queue 1, class 2 is associated with queue 2, and so on.

A class can be associated with a different queue on each interface. For example, in Figure 7–3, class 3 is associated with queue 3 on the W622-5-0 interface and with queue 1 on the W622-5-1 interface.

It is the queue which determines the actual priority of the packets. Any packets on queue 1 are transmitted first, followed by queue 2, and so on.

In Figure 7–3, queue 1 on the W622-5-0 interface contains "telnet" and "br_lat" packets, and therefore these are transmitted before any "decnet", "ftp" and "mop" packets.

More than one class can be associated with the same queue. If this occurs, each class has equal priority. For example, in Figure 7–3, class 1 and class 3 are both associated with queue 1 on W622-5-1. This gives all the packets the same priority.

7.6 Which Packet Types Can Be Prioritized?

You can prioritize all types of packet, apart from control packets and X.25 packets. The DECNIS recognizes five categories of packet:

- IP
- Bridge
- DECnet Phase IV
- OSI
- IPX (only on L601 or L602)

You can use PACKET entities and/or the DEFAULT CLASS characteristics to prioritize IP, bridge and DECnet Phase IV packets

You can only use the DEFAULT CLASS characteristics to prioritize OSI and IPX packets.

Section 7.6.1 describes the DEFAULT CLASS characteristics. Section 7.6.4 to Section 7.6.6 describe setting up PACKET entities for IP, bridge and DECnet Phase IV packets.

7.6.1 DEFAULT CLASS Characteristics

The default classes are a characteristic of the PRIORITY module. If the DECNIS receives a packet and there is no matching PACKET entity, the DECNIS associates the packet with a DEFAULT CLASS characteristic.

There are ten DEFAULT CLASS characteristics:

- IP TOS DEFAULT CLASS
- IP TCP DEFAULT CLASS
- IP UDP DEFAULT CLASS
- IP PROTOCOL DEFAULT CLASS
- BRIDGE SAP DEFAULT CLASS

- BRIDGE PID DEFAULT CLASS
- BRIDGE TYPE DEFAULT CLASS
- DECNET PHASE IV DEFAULT CLASS
- OSI DEFAULT CLASS
- IPX DEFAULT CLASS

The DEFAULT CLASS characteristics (apart from IP TOS) are automatically associated with class 4. Class 4 is automatically associated with queue 4, and therefore any packets associated with a DEFAULT CLASS characteristic automatically have a medium priority.

To change the priority of a DEFAULT CLASS characteristic, you associate it with a different class. Section 7.7.6 describes the NCL command which associates a class with a DEFAULT CLASS.

The IP TOS DEFAULT CLASS characteristic is not associated with a class but is set to "Ignore TOS field" (see Section 7.6.4 for details about prioritizing with TOS (Type of Service) values).

7.6.2 OSI Packets

You cannot use PACKET entities to set the priority of OSI packets. You must use DEFAULT CLASS characteristics to set the priority.

Note: Be careful that you do not set the OSI priority too low. If you set up a situation where there is a lot of high-priority traffic and OSI is a low priority, some OSI packets may not get through to the DECNIS. This could mean that you are unable to manage the DECNIS using NCL.

7.6.3 IPX Packets

You cannot use PACKET entities to set the priority of IPX packets. You must use DEFAULT CLASS characteristics to set the priority.

Note: the DECNIS only supports Native IPX packets on the L601 and L602 Network Interface Cards.

7.6.4 IP Packets

To set up a PACKET entity with an IP packet description, you can use any of the following packet characteristics:

- TOS (Type of Service) value
- TCP/UDP port number
- Protocol type

For example, to describe a Telnet packet you would use the TCP Port number 23.

When the DECNIS receives an IP packet, it carries out a three stage check of the packet characteristics. Figure 7–4 shows the three stages.

Stage 1

The DECNIS looks at the TOS value. If the TOS value is zero, it carries out the Stage 2 check.

If the TOS value is greater than zero, the DECNIS looks for a PACKET entity with a matching TOS value and takes the MATCH ACTION of that entity. If there is no PACKET entity with the same TOS value, the DECNIS looks at the IP TOS DEFAULT CLASS characteristic. If this is set to "Ignore TOS Field", the DECNIS carries out the Stage 2 check, otherwise it assigns the packet to the IP TOS DEFAULT CLASS.

Stage 2

The DECNIS checks whether the packet contains a TCP or UDP port number. (If a TCP or UDP packet has been fragmented then the port number is only contained in the first fragment of the packet.)

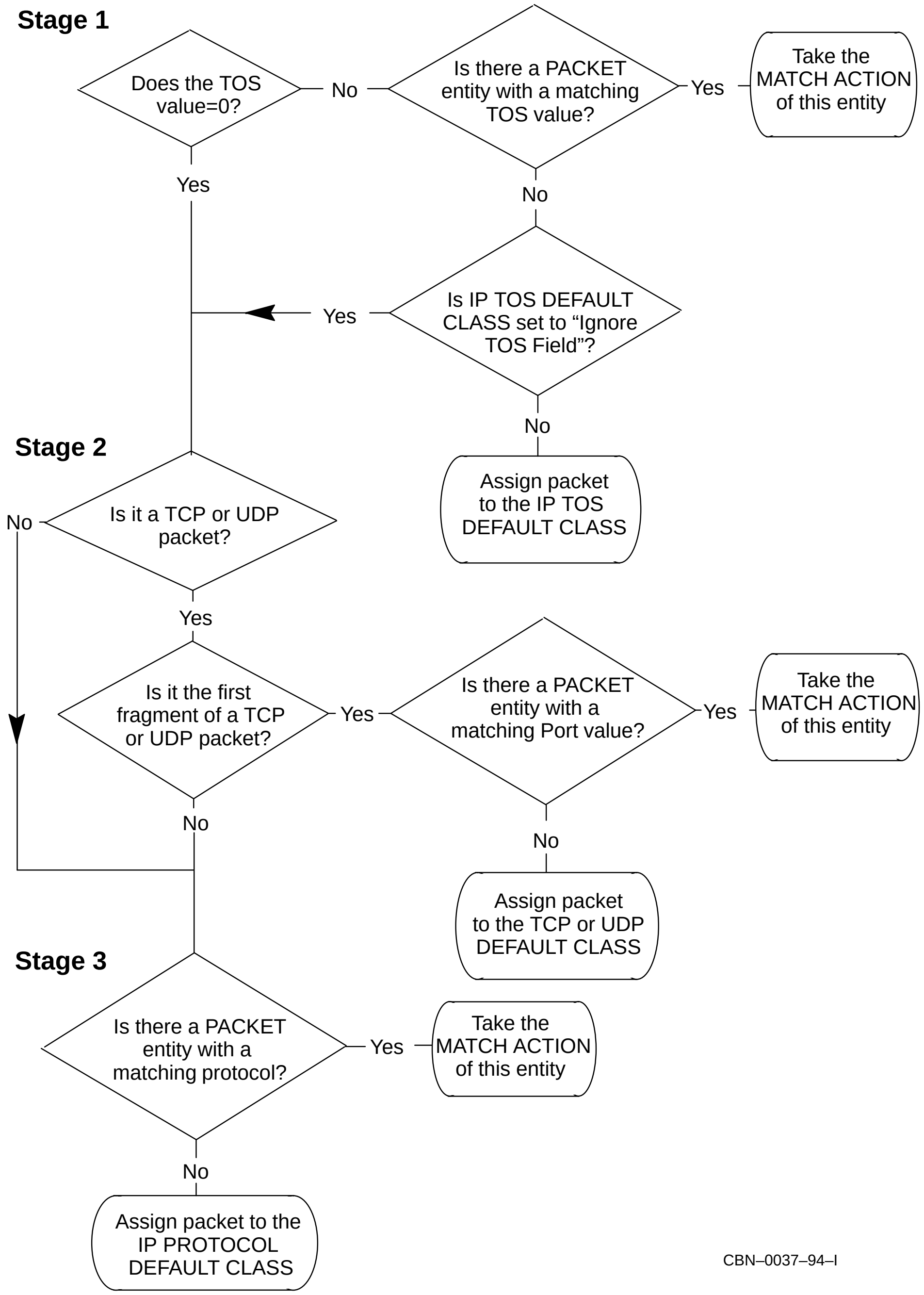
If the packet contains a TCP/UDP port number, the DECNIS looks for a PACKET entity with a matching port number and takes the MATCH ACTION of that entity. If there is no PACKET entity with the same port number, the DECNIS assigns it to the TCP or UDP DEFAULT CLASS.

If it does not contain a TCP/UDP port number, the DECNIS carries out the Stage 3 check.

Stage 3

The DECNIS looks at the protocol type of the packet and checks for a matching PACKET entity. If there is no PACKET entity with the same protocol, the DECNIS assigns the packet to the IP PROTOCOL DEFAULT CLASS.

Figure 7–4 Prioritizing IP Packets


Stage 1
Does the TOS value=0?
No
Is there a PACKET entity with a matching TOS value?
Yes
Take the MATCH ACTION of this entity
Yes
No
Yes
Is IP TOS DEFAULT CLASS set to "Ignore TOS Field"?
No
Assign packet to the IP TOS DEFAULT CLASS
Stage 2
No
Is it a TCP or UDP packet?
Yes
Is it the first fragment of a TCP or UDP packet?
Yes
Is there a PACKET entity with a matching Port value?
Yes
Take the MATCH ACTION of this entity
No
No
Assign packet to the TCP or UDP DEFAULT CLASS
Stage 3
Is there a PACKET entity with a matching protocol?
Yes
Take the MATCH ACTION of this entity
No
Assign packet to the IP PROTOCOL DEFAULT CLASS
CBN–0037–94–I

7.6.4.1 The IP TOS DEFAULT CLASS

The IP TOS DEFAULT CLASS characteristic is automatically set to "Ignore TOS Field". This means that if the DECNIS receives a packet with a TOS value set and there is no PACKET entity with a matching TOS value, it will not assign the packet to the IP TOS DEFAULT CLASS. Instead, the DECNIS will continue looking for a PACKET entity that matches the protocol or port characteristics of the packet.

For example, if the DECNIS receives a TCP packet with a TOS value set, it looks for a PACKET with a matching TOS value. If there is no matching PACKET entity and the IP TOS DEFAULT CLASS is set to "Ignore TOS Field", the DECNIS looks for a PACKET entity with a matching TCP port number.

7.6.4.2 TOS Values

The TOS value is used to indicate how the packet should be treated during its transmission through the internet system. The DECNIS supports TOS values as outlined in RFC 791.

The TOS field contains a Precedence subfield and a Service subfield. The DECNIS checks both subfields unless you specify "ignore" for one of the subfields. You would specify "ignore" if only one field is important. For example, if you want the DECNIS to prioritize "any" packet requesting "minimize delay", you specify "ignore" for the Precedence subfield and "minimize delay" for the Service subfield.

The Precedence Subfield

The Precedence subfield uses the type of packet to indicate the importance of the packet. The types it uses are:

- network control
- internetwork control
- critic
- flash override
- flash
- immediate
- priority
- routine
- ignore

where "network control" is the most important type and "routine" is the least important type.

The Service Subfield

The Service subfield indicates the most important factor to be considered when the packet is transmitted over a network:

- minimize delay
- maximize throughput
- maximize reliability
- minimize monetary cost
- normal service
- ignore

7.6.4.3 Protocol Types

Each protocol type has an assigned number. Table 7–1 lists some commonly used protocol types and their assigned numbers.

Table 7–1 Protocol Numbers

Protocol Type	Protocol Number	Protocol Name
EGP	8	Exterior Gateway Protocol
ICMP	1	Internet Control Message
IGMP	2	Internet Group Management
TCP	6	Transmission Control
UDP	17	User Datagram

For a complete listing of protocol numbers, refer to RFC 1700 (October 1994).

7.6.4.4 TCP/UDP Port Numbers

Some services and protocols are linked to specific TCP and UDP port numbers. Table 7–2 lists some commonly used services and protocols and their port numbers.

Table 7–2 TCP/UDP Port Numbers

Service Type	Port Number	Service Name
AT-NBP	202	AppleTalk Name Binding
AT-RTMP	201	AppleTalk Routing Maintenance
AT-ZIS	206	AppleTalk Zone Information
BGP	179	Border Gateway Protocol
BOOTPC	68	Bootstrap Protocol Client
BOOTPS	67	Bootstrap Protocol Server
FTP	21	File Transfer [Control]
FTP-DATA	20	File Transfer [Default Data]
GOPHER	70	Gopher
IPX	213	IPX
LOGIN	49	Login Host Protocol
NETBIOS-DGM	138	NETBIOS Datagram Service
NETBIOS-NS	137	NETBIOS Name Service
NETBIOS-SSN	139	NETBIOS Session Service
RTELNET	107	Remote Telnet Service
SMTP	25	Simple Mail Transfer
SNAGAS	108	SNA Gateway Access Server
SNMP	161	SNMP
SNMPTRAP	162	SNMP TRAP
TELNET	23	Telnet
TFTP	69	Trivial File Transfer

(continued on next page)

Table 7–2 (Cont.) TCP/UDP Port Numbers

Service Type	Port Number	Service Name
WWW	80	World Wide Web HTTP
XDMCP	177	X Display Manager Control Protocol

For a complete listing of port numbers, refer to RFC 1700 (October 1994).

7.6.4.5 Encapsulated AppleTalk and IPX Packets

You can prioritize any AppleTalk and NetWare IPX packets that are encapsulated within IP packets.

To set up a PACKET entity with an encapsulated AppleTalk packet description, you specify a UDP Port number of 748.

To set up a PACKET entity with an encapsulated IPX packet description, you specify a UDP Port number of 213.

7.6.5 Bridge Packets

To set up a PACKET entity with a bridge packet description, you can use any of the following packet characteristics:

- SAP value
- PID value
- Ethernet type

For example, to describe a LAT packet you would use the Ethernet type 60-04.

7.6.6 DECnet Phase IV Packets

To set up a PACKET entity with a DECnet Phase IV packet description, you use the packet length.

For example, you could use the packet length 1..30 to describe small packets.

7.7 Tasks Required to Set Up Packet Prioritization

To set up prioritization, you need to carry out the tasks shown in Figure 7–6. Section 7.7.1 to Section 7.7.10 describe the NCL commands required to perform these tasks.

You can use the template in Figure 7–5 to help you plan your prioritization needs.

Figure 7–5 Prioritization Template

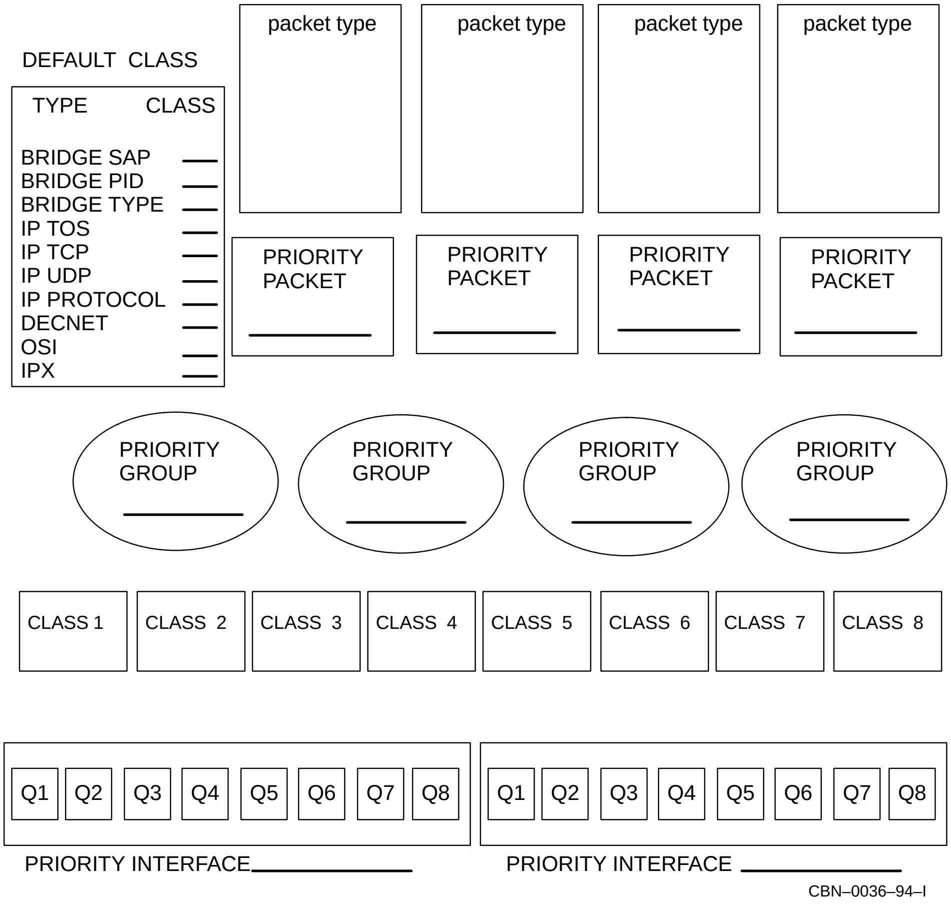
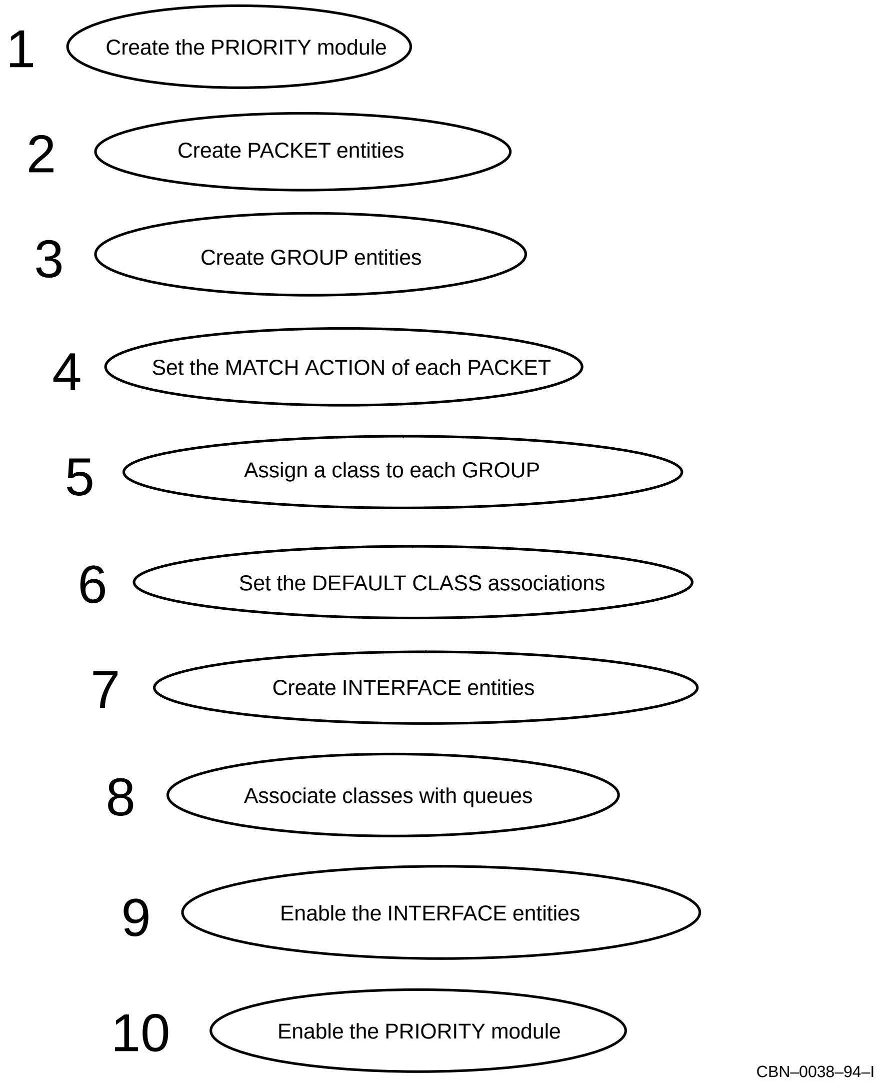


Figure 7–6 Tasks


1
Create the PRIORITY module
2
Create PACKET entities
3
Create GROUP entities
4
Set the MATCH ACTION of each PACKET
5
Assign a class to each GROUP
6
Set the DEFAULT CLASS associations
7
Create INTERFACE entities
8
Associate classes with queues
9
Enable the INTERFACE entities
10
Enable the PRIORITY module
CBN–0038–94–I

7.7.1 Create the PRIORITY Module

To create the PRIORITY module, enter the following command:

```
NCL> CREATE PRIORITY
```

Note: if the PRIORITY module has already been created for IP packet filtering, then the PRIORITY module must be Uncoupled (see Section 7.9.1).

7.7.2 Create PACKET Entities

You should create a PACKET entity for each type of packet you want to prioritize (apart from OSI and IPX). Any types without a PACKET entity will be assigned to a DEFAULT CLASS: see Section 7.6.

To create a PACKET entity, enter the following command:

```
NCL> CREATE PRIORITY PACKET packet-name -
_NCL> TYPE type, characteristic = char-value
```

where:

packet-name is a name you choose to represent the type of packet

type is one of the following packet types:

- IP TOS
- IP TCP
- IP UDP
- IP PROTOCOL
- BRIDGE SAP
- BRIDGE PID
- BRIDGE TYPE
- DECNET

(For more details about packet types, see Section 7.6.)

characteristic is one of the characteristics from Table 7–3

char-value is the value for the characteristic, in the format shown in Table 7–3

Example

To create a PACKET entity for Telnet packets, enter the following command:

```
NCL> CREATE PRIORITY PACKET telnet TYPE ip tcp , ip port range = [23..23]
```

Table 7–3 PACKET Entity Characteristics

Packet Type	*Characteristic*	*char-value* Format	Default *char-value*
IP TOS	IP TOS VALUE	{PRECEDENCE = precedence[1], SERVICE = service[1]}	{PRECEDENCE=routine, SERVICE=minimize monetary cost}
IP TCP	IP PORT RANGE	[*n*..*n*]	[0..0]
IP UDP	IP PORT RANGE	[*n*..*n*]	[0..0]
IP PROTOCOL	IP PROTOCOL	*n*	0
BRIDGE SAP	BRIDGE SAP VALUE	*nn*	00
BRIDGE PID	BRIDGE PID VALUE	*nn-nn-nn-nn-nn*	00-00-00-00-00
BRIDGE TYPE	BRIDGE ETHERNET PROTOCOL TYPE	*nn-nn*	00-00
DECNET	PACKET LENGTH	[*n*..*n*]	[0..0]

[1] See Section 7.6.4.2 for precedence and service types.

7.7.3 Create GROUP Entities

Each packet must be associated with a GROUP entity.

To create a GROUP entity, enter the following command:

```
NCL> CREATE PRIORITY GROUP group-name
```

where *group-name* is a name you choose. It is useful to use a name which represents the type of packets in the group. For example, you could use the name "interactive" to represent all interactive traffic such as Telnet and LAT.

Example

To create a GROUP entity called "interactive", enter the following command:

```
NCL> CREATE PRIORITY GROUP interactive
```

7.7.4 Set the Match Action of Each PACKET Entity

Each PACKET entity must have a MATCH ACTION which associates it with a group. To set the MATCH ACTION, enter the following command:

```
NCL> SET PRIORITY PACKET packet-name MATCH ACTION = PRIORITY GROUP group-name
```

where:

packet-name	is the name of a PACKET entity
group-name	is the name of a GROUP entity

Example

To set the MATCH ACTION to associate a PACKET entity called "telnet" with a GROUP entity called "interactive", enter the following command:

```
NCL> SET PRIORITY PACKET telnet MATCH ACTION = PRIORITY GROUP interactive
```

7.7.5 Assign a Class to Each GROUP Entity

Each GROUP entity must be assigned a class, from class 1 to class 8. Enter the following command:

```
NCL> SET PRIORITY GROUP group-name ASSIGNED CLASS class-number
```

where:

group-name	is the name of a GROUP entity
class-number	is a number from 1 to 8

Example

To assign class 1 to the GROUP "interactive", enter the following command:

```
NCL> SET PRIORITY GROUP interactive ASSIGNED CLASS 1
```

7.7.6 Set the DEFAULT CLASS Associations

The DEFAULT CLASS characteristics (apart from IP TOS) are automatically associated with class 4.

To set the DEFAULT CLASS association, enter the following command:

```
NCL> SET PRIORITY def-class class-number
```

where:

def-class is one of the ten DEFAULT CLASSes:

- IP TOS DEFAULT CLASS
- IP TCP DEFAULT CLASS
- IP UDP DEFAULT CLASS
- IP PROTOCOL DEFAULT CLASS
- BRIDGE SAP DEFAULT CLASS
- BRIDGE PID DEFAULT CLASS
- BRIDGE TYPE DEFAULT CLASS
- DECNET PHASE IV DEFAULT CLASS
- OSI DEFAULT CLASS
- IPX DEFAULT CLASS

class-number is the class number (1 to 8) you want to associate with the DEFAULT CLASS

Example

To set the OSI DEFAULT CLASS to class 3, enter the following command:

```
NCL> SET PRIORITY osi default class 3
```

Note 1: Setting the DEFAULT CLASS association is the only way to prioritize OSI and IPX packets.

Note 2: The IP TOS DEFAULT CLASS is automatically associated with "Ignore TOS Field". Do not change the IP TOS DEFAULT CLASS from "Ignore TOS Field", unless your whole prioritization scheme is based upon TOS values (see Section 7.6.4).

7.7.7 Create INTERFACE Entities

Create an INTERFACE entity for each communications port you want to use prioritization on. Enter the following command:

```
NCL> CREATE PRIORITY INTERFACE interface-name COMMUNICATION PORT comm-port
```

where:

interface-name is a name to represent the interface

comm-port is the name of the communications port

Note: you can only use prioritization on the W622, W618, W614, L601 and L602 Network Interface Cards.

Example

To set up an INTERFACE entity for W622-5-0, enter the following command:

```
NCL> CREATE PRIORITY INTERFACE w622-5-0 COMMUNICATION PORT w622-5-0
```

7.7.8 Associate Classes with Queues

By default, each class is associated with its corresponding queue, that is, class 1 with queue 1, class 2 with queue 2, and so on. If you want to change these associations, enter the following command:

```
NCL> SET PRIORITY INTERFACE interface-name CLASS class-number ASSIGNED -
_NCL> QUEUE = queue-number
```

where:

interface-name is the name of the interface on which you are changing the class association

class-number is the class number

queue-number is the queue number

Example

To associate class 1 with queue 2 on W622-5-0, enter the following command:

```
NCL> SET PRIORITY INTERFACE w622-5-0 CLASS 1 ASSIGNED QUEUE = 2
```

7.7.9 Enable the INTERFACE entities

Enable each PRIORITY INTERFACE by entering the following command:

```
NCL> ENABLE PRIORITY INTERFACE interface-name
```

Example

To enable the W622-5-0 interface, enter the following command:

```
NCL> ENABLE PRIORITY INTERFACE w622-5-0
```

7.7.10 Enable the PRIORITY Module

When you have completed all the previous tasks, you must enable the PRIORITY module.

To enable the PRIORITY module, enter the following command:

```
NCL> ENABLE PRIORITY
```

7.8 Examples of Setting Up Packet Prioritization

This section shows the NCL commands required to set up the packet prioritization shown in Figure 7–7 and Figure 7–8. Some of these NCL commands are included in example script files. From the example script files, you can copy the NCL commands you require into the user NCL script files for your DECNIS. For more details about the user NCL script files, refer to Section 1.5.6.

For details about the location of the example script files on various load hosts, refer to Appendix G.

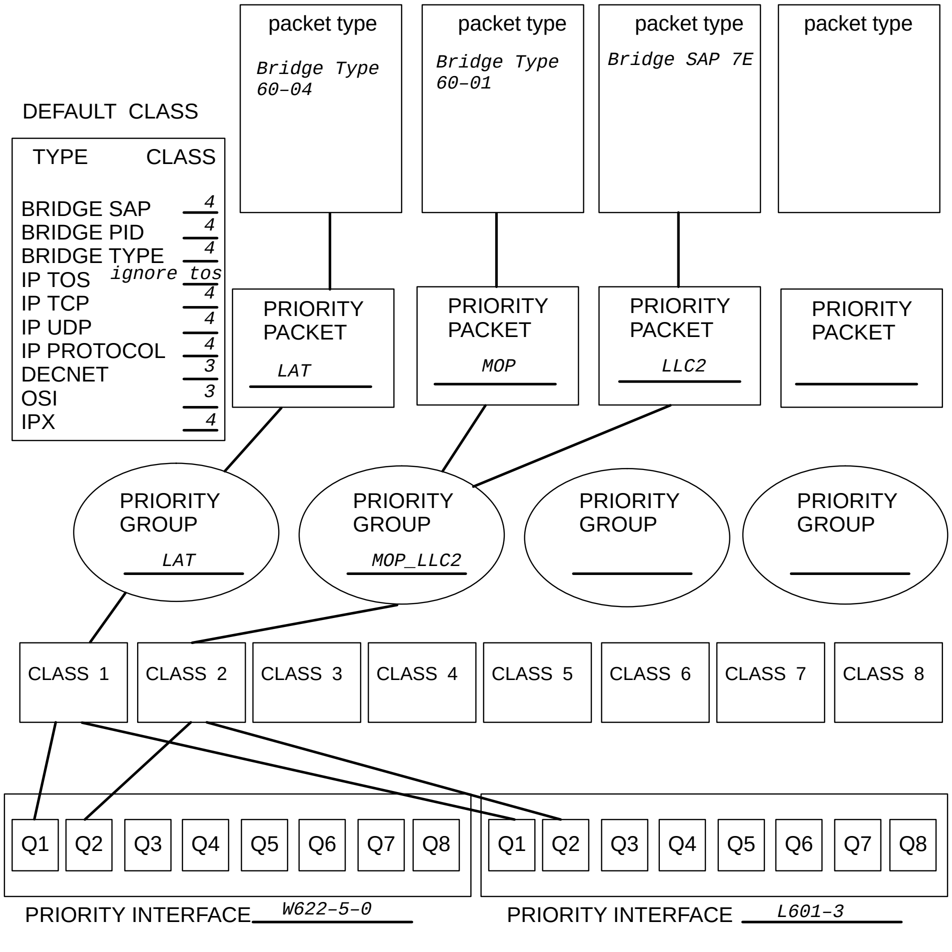
Example One

This shows an example of prioritization in a mainly bridging environment.

```
NCL> CREATE PRIORITY

NCL> CREATE PRIORITY PACKET lat TYPE bridge type , bridge ethernet -
_NCL> protocol type = 60-04
NCL> CREATE PRIORITY PACKET mop TYPE bridge type , bridge ethernet -
_NCL> protocol type = 60-01
NCL> CREATE PRIORITY PACKET llc2 TYPE bridge sap , bridge sap value = 7E

NCL> CREATE PRIORITY GROUP lat
NCL> CREATE PRIORITY GROUP mop_llc2

NCL> SET PRIORITY PACKET lat MATCH ACTION = PRIORITY GROUP lat
NCL> SET PRIORITY PACKET mop MATCH ACTION = PRIORITY GROUP mop_llc2
NCL> SET PRIORITY PACKET llc2 MATCH ACTION = PRIORITY GROUP mop_llc2

NCL> SET PRIORITY GROUP lat ASSIGNED CLASS 1
NCL> SET PRIORITY GROUP mop_llc2 ASSIGNED CLASS 2

NCL> SET PRIORITY DECNET PHASEIV DEFAULT CLASS 3
NCL> SET PRIORITY OSI DEFAULT CLASS 3


NCL> CREATE PRIORITY INTERFACE w622-5-0 COMMUNICATION PORT w622-5-0
NCL> CREATE PRIORITY INTERFACE l601-3 COMMUNICATION PORT l601-3

NCL> ENABLE PRIORITY INTERFACE w622-5-0
NCL> ENABLE PRIORITY INTERFACE l601-3

NCL> ENABLE PRIORITY
```

Figure 7–7 Example One

Priority	Packet Type
1	*LAT*
2	MOP, LLC2
3	*Any DECnet, any OSI*
4	*Any other bridged, IP or IPX*



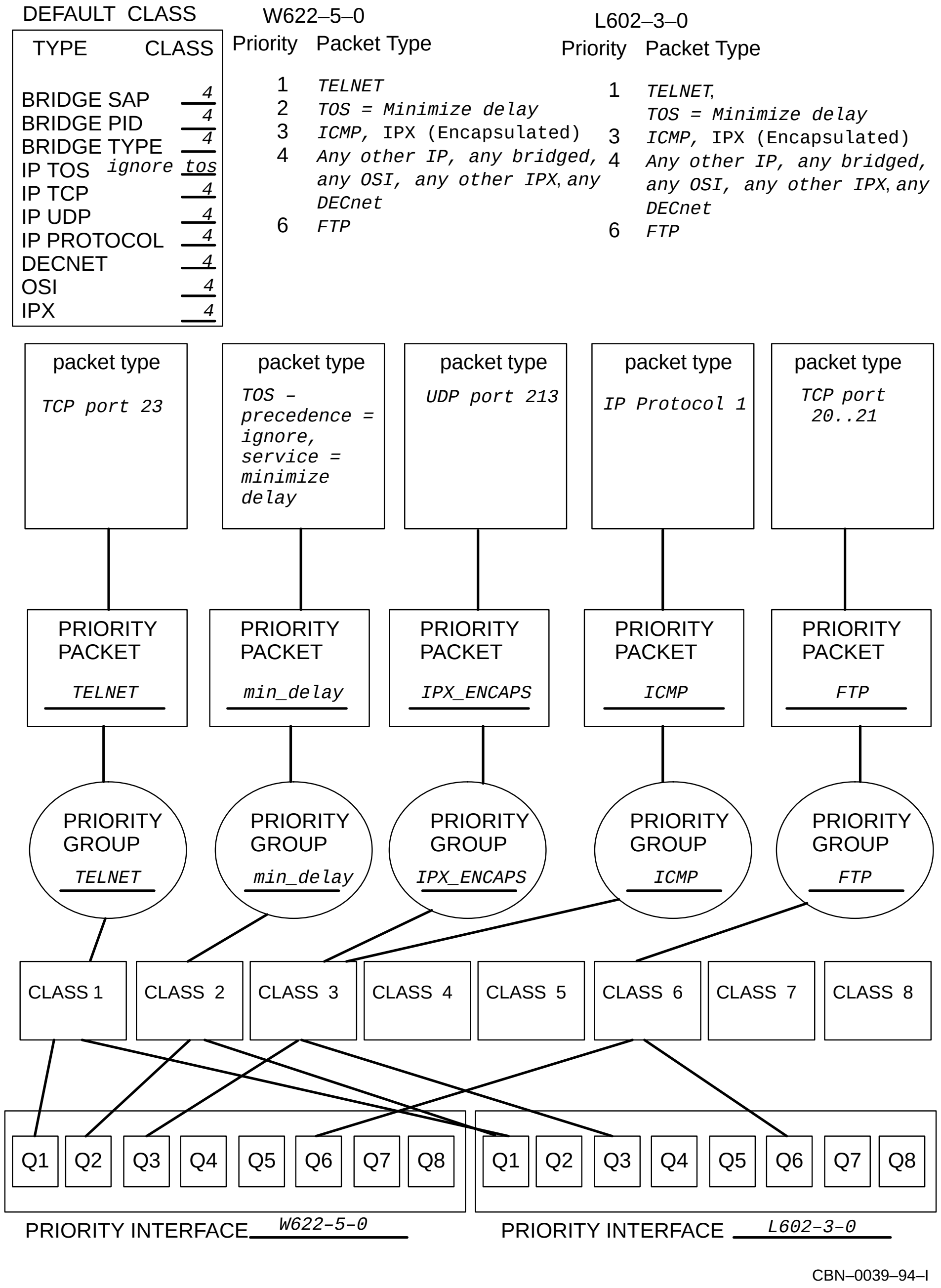
CBN–0040–94–I

Example Two

This shows an example of prioritization in a mainly IP environment.

```
NCL> CREATE PRIORITY

NCL> CREATE PRIORITY PACKET telnet TYPE ip tcp , ip port range = [23..23]
NCL> CREATE PRIORITY PACKET min_delay TYPE ip tos , ip tos -
_NCL> value = {precedence = ignore , service = minimize delay}

NCL> CREATE PRIORITY PACKET ipx_encaps TYPE ip udp , -
_NCL> ip port range = [213..213]

NCL> CREATE PRIORITY PACKET icmp TYPE ip protocol , ip protocol = 1
NCL> CREATE PRIORITY PACKET ftp TYPE ip tcp , ip port range = [20..21]

NCL> CREATE PRIORITY GROUP telnet
NCL> CREATE PRIORITY GROUP min_delay
NCL> CREATE PRIORITY GROUP ipx_encaps
NCL> CREATE PRIORITY GROUP icmp
NCL> CREATE PRIORITY GROUP ftp

NCL> SET PRIORITY PACKET telnet MATCH ACTION = PRIORITY GROUP telnet
NCL> SET PRIORITY PACKET min_delay MATCH ACTION = PRIORITY GROUP min_delay
NCL> SET PRIORITY PACKET ipx_encaps MATCH ACTION = PRIORITY GROUP ipx_encaps
NCL> SET PRIORITY PACKET icmp MATCH ACTION = PRIORITY GROUP icmp
NCL> SET PRIORITY PACKET ftp MATCH ACTION = PRIORITY GROUP ftp

NCL> SET PRIORITY GROUP telnet ASSIGNED CLASS 1
NCL> SET PRIORITY GROUP min_delay ASSIGNED CLASS 2
NCL> SET PRIORITY GROUP ipx_encaps ASSIGNED CLASS 3
NCL> SET PRIORITY GROUP icmp ASSIGNED CLASS 3
NCL> SET PRIORITY GROUP ftp ASSIGNED CLASS 6

NCL> CREATE PRIORITY INTERFACE w622-5-0 COMMUNICATION PORT w622-5-0

NCL> CREATE PRIORITY INTERFACE l602-3-0 COMMUNICATION PORT l602-3-0

NCL> SET PRIORITY INTERFACE l602-3-0 CLASS 2 ASSIGNED QUEUE = 1

NCL> ENABLE PRIORITY INTERFACE w622-5-0
NCL> ENABLE PRIORITY INTERFACE l602-3-0

NCL> ENABLE PRIORITY
```

Figure 7–8 Example Two



7.9 Changing Your Prioritization Settings

Whilst the PRIORITY module is enabled, you can change:

- The MATCH ACTION of PACKET entities
- The ASSIGNED CLASS number of a GROUP entity
- The class-to-queue association on an INTERFACE entity

If you need to make any other changes to your prioritization set up, such as creating a new PACKET entity, you must Uncouple the PRIORITY module.

7.9.1 The Uncouple and Update Commands

You can modify your prioritization settings without affecting the current operation of any existing prioritization settings or any IP packet filters.

Follow these steps:

1. Uncouple the PRIORITY module by entering the following command:

   ```
   NCL> UNCOUPLE PRIORITY
   ```

2. Modify your prioritization settings, using NCL Set commands.
3. Replace the existing prioritization settings with the new prioritization settings by issuing the Update command:

   ```
   NCL> UPDATE PRIORITY
   ```

7.10 Optimizing Packet Prioritization

Once you have set up packet prioritization, you can monitor how it affects the traffic flow through the DECNIS. This is described in Section 7.10.1.

You can make the flow of packets more efficient by assigning packets to different queues or adjusting the maximum queue length characteristic: see Section 7.10.2.

7.10.1 Monitoring Traffic

You can monitor the amount of traffic on each interface by looking at the counters for each QUEUE. Enter the following command:

```
NCL> SHOW PRIORITY INTERFACE interface-name QUEUE queue-name ALL COUNTERS
```

This command displays the following information:

- Number of data packets transmitted
- Number of data bytes transmitted
- Number of data packets discarded

Example

To monitor the flow of traffic on all queues on interface W622-5-0, enter the following command:

```
NCL> SHOW PRIORITY INTERFACE w622-5-0 QUEUE * ALL COUNTERS
```

From looking at the number of packets transmitted, you might determine that a particular type of packet is more common than you expected. In this situation, you may want to make those packets a higher priority. For example, you may determine that there are a large amount of file transfers taking place, so that you want to make FTP packets a higher priority.

From looking at the number of packets discarded, you might decide that too many packets are being discarded. In this situation, you could increase the maximum queue length for the queue. If more than one GROUP is associated with that queue, you could associate one of the GROUPs with another queue.

7.10.2 Setting the Maximum Queue Length

Each queue contains packets waiting for transmission. The MAXIMUM QUEUE LENGTH value is the number of packets allowed on the queue at one time. If the number of packets on the queue equals the maximum value, then any further packets assigned to the queue are discarded.

The default MAXIMUM QUEUE LENGTH values are:

- queue 1, 65535 packets
- queue 2 to queue 8, 100 packets

The MAXIMUM QUEUE LENGTH can be set to any value up to 65535.

To set the MAXIMUM QUEUE LENGTH, enter the following command:

```
NCL> SET PRIORITY INTERFACE interface-name QUEUE queue-number -
_NCL> MAXIMUM QUEUE LENGTH number-of-packets
```

where *number-of-packets* is the number of packets allowed to wait on the queue.

Example

To set the MAXIMUM QUEUE LENGTH to 100 on queue 8, enter the following command:

```
NCL> SET PRIORITY INTERFACE w622-5-0 QUEUE 8 -
_NCL> MAXIMUM QUEUE LENGTH = 100
```

You may want to set the MAXIMUM QUEUE LENGTH of high-priority queues to a high value. This ensures that high-priority packets are not discarded. It is advisable to set the MAXIMUM QUEUE LENGTH of low-priority queues to a lower value. This ensures that the DECNIS does not fill all its buffers with low-priority traffic.

7.10.2.1 Discarding Packets

If you set the MAXIMUM QUEUE LENGTH of a queue to zero, any packets assigned to that queue are discarded.

If there are specific packets that you want to discard, assign them to a queue with a MAXIMUM QUEUE LENGTH of zero. For example, you could discard all DECnet packets by assigning them to queue 8, and then setting the MAXIMUM QUEUE LENGTH of queue 8 to zero.

The examples in Section 7.12.5 show how the MAXIMUM QUEUE LENGTH can be used to discard packets.

7.11 Using Pattern Matching to Extend Packet Prioritization

This section describes the pattern matching extension to packet prioritization.

7.11.1 What Is Pattern Matching?

Pattern matching is an extension to packet prioritization that lets you be more specific about the packets you want to prioritize.

For example, with basic prioritization, you can specify that IP Telnet packets have a high priority. With pattern matching, you could specify that only IP Telnet packets containing a specific byte sequence have a high priority.

The byte sequence might represent any information in the packet, for example:

- Destination address
- Source address
- Service type
- Subnet address

For example, you could specify that all Telnet packets with a destination address of 16.31.16.16 have a high priority.

7.11.2 How Pattern Matching Works

With basic prioritization, you identify packets using a PACKET entity containing a description of a packet. For example, to identify Telnet packets, the PACKET entity contains the IP TCP PORT number 23.

With pattern matching, you use the same PACKET entity and associate it with a PATTERN entity.

7.11.2.1 The PATTERN Entity

The PATTERN entity is a subentity of the PRIORITY module. The PATTERN entity describes:

- A byte sequence
- A mask for the byte sequence
- The location of the byte sequence in the packet

Byte Sequence

The byte sequence is a string of HEX digits. The maximum length of a byte sequence is 32 bytes. For example, a destination address of 16.31.16.16 is represented by the following hex digits: 101F1010.

Mask

The mask is a string of HEX digits which represents a binary mask. The mask must be the same length as the byte sequence.

The mask indicates which digits in the byte sequence are significant. For example, the byte sequence 101F1010 could have a mask of FFFFFFFF. This means that all the digits are significant, and therefore only packets with exactly the same destination address will match this PATTERN entity.

If the mask was FFFFFFF0, then the last hex digit is not significant. This means that the last pair of hex digits can be any from 10 to 1F, and therefore any packets with a destination address from 16.31.16.16 to 16.31.16.31 will match this PATTERN entity.

Location

To find the correct byte sequence in the packet, the DECNIS needs to know the offset of the byte sequence. The offset is the total number of bytes preceding the byte sequence.

To work out the offset, you need to know the frame format of the packet and then count how many bytes occur before the byte sequence.

You can also specify whether the DECNIS counts the offset from the START of the packet or from the NEXT ISO LAYER in the packet.

It is useful to specify NEXT ISO LAYER if you are bridging over different media (for example, Ethernet and FDDI) because the packet format will be different depending on the media.

The maximum offset is 62. A byte sequence must not extend beyond 63 bytes into the packet.

Figure 7–9 shows an example frame format of an IP packet.

In Figure 7–9, to describe the location of the IP Destination Address, you would specify an offset of 16 from the START of the packet.

Figure 7–10 shows an example of an IPX packet in a bridged Ethernet format frame. Figure 7–11 shows an example of an IPX packet in a bridged 802.3 format frame.

In both Figure 7–10 and Figure 7–11, to describe the location of the IPX Destination Network, you would specify an offset of 6 from the NEXT ISO LAYER.

Figure 7–9 Offset in an IP Frame

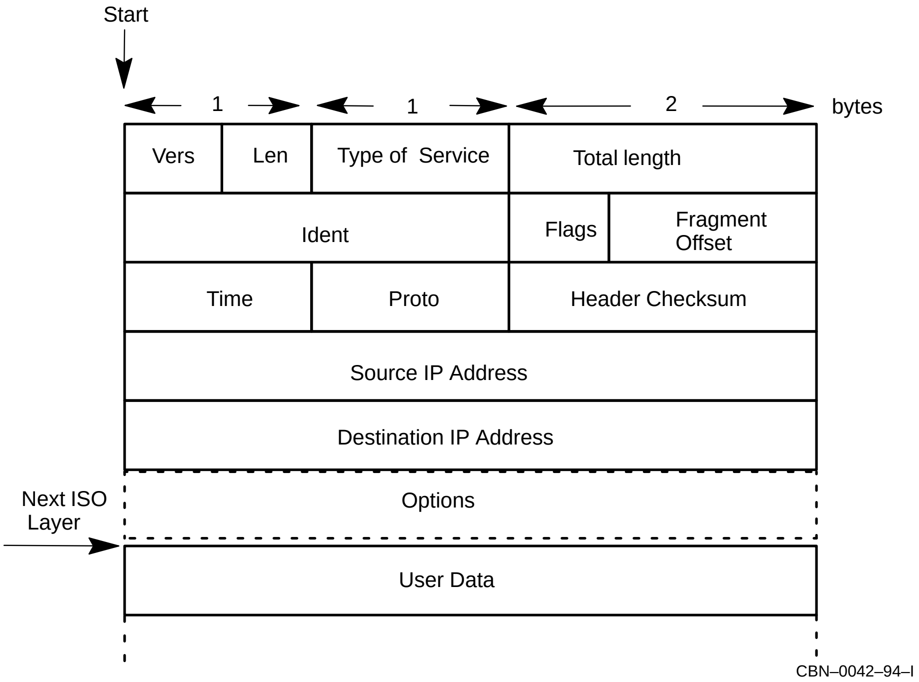


Figure 7–10 Offset in a Bridged IPX Ethernet Format Frame

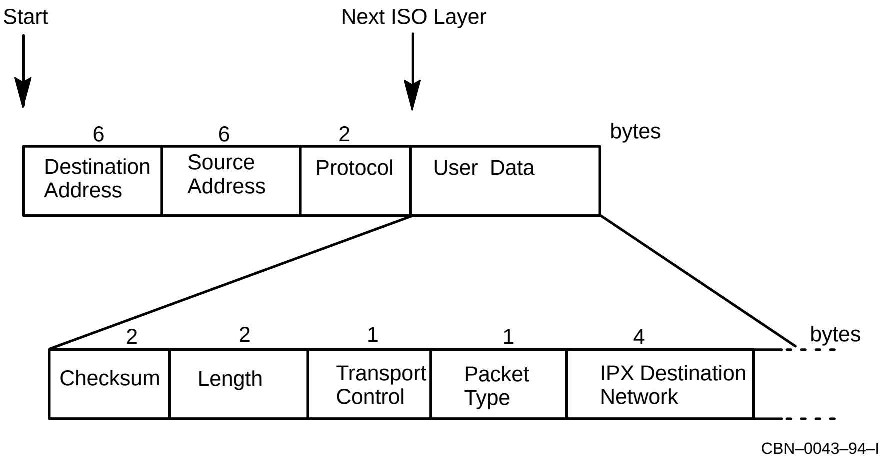
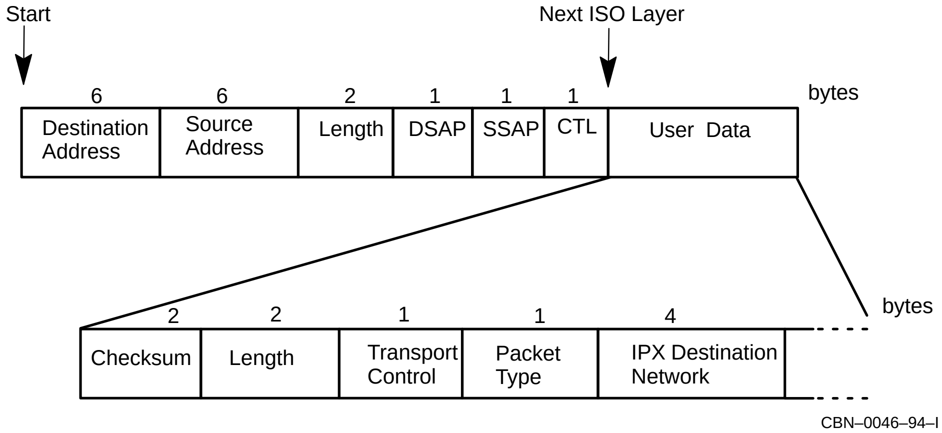


Figure 7–11 Offset in a Bridged IPX 802.3 Format Frame



7.11.3 How Pattern Matching Fits in with Basic Prioritization

Figure 7–12 shows how the PATTERN entity fits in with the other PRIORITY entities. The example in Figure 7–12 is taken from Figure 7–3, with the addition of pattern matching for Telnet packets.

1. As in basic prioritization, when the DECNIS receives a packet it checks all the PACKET entities for a description that matches the packet type.

   If the DECNIS finds a matching PACKET entity, it takes the "match action" associated with the entity.

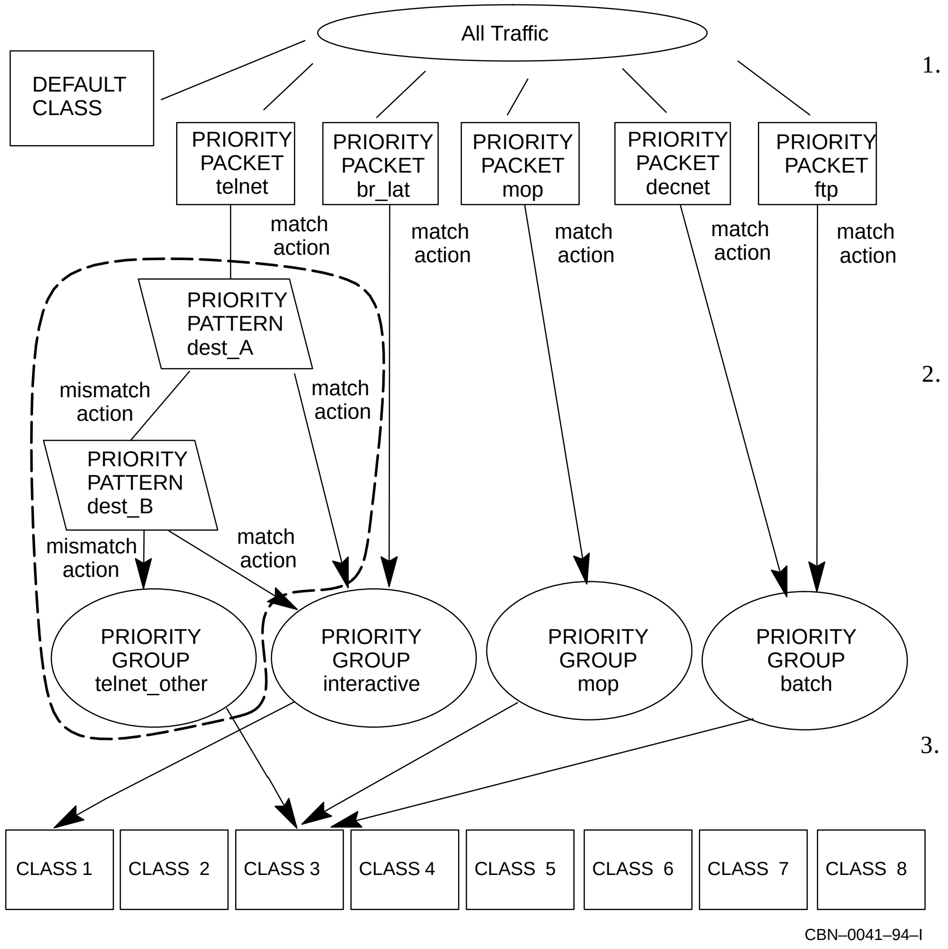
   If it does not find a matching PACKET entity, it associates the packet with a DEFAULT CLASS characteristic.

   With pattern matching, the "match action" associates the packet with a PATTERN entity (rather than a GROUP, as in basic prioritization).

2. The PATTERN entities describe a specific byte sequence and mask, at a specific position in the packet.

   If the packet matches the description in the PATTERN entity, then the DECNIS takes the "match action" associated with the PATTERN entity. The match action associates the packet with either another PATTERN entity, or a GROUP entity.

Figure 7–12 How Pattern Matching Fits in with Basic Prioritization



If the packet does not match the description in the PATTERN entity, then the DECNIS takes the "mismatch action" associated with the PATTERN entity. The mismatch action associates the packet with either another PATTERN entity, or a GROUP entity.

Note: If there is no mismatch action associated with the PATTERN entity, then the DECNIS associates the packet with a DEFAULT CLASS characteristic.

3. As in basic prioritization, each GROUP entity is associated with a class. Classes are then associated with queues.

In Figure 7–12, pattern matching is set up for Telnet packets. Any Telnet packets that the DECNIS receives are assigned to a PACKET entity called "telnet". All packets assigned to "telnet" are associated with the PATTERN entity "dest_A".

The PATTERN entity "dest_A" contains a byte sequence representing destination address A. If the packets contain destination address A, they are assigned to GROUP entity "interactive".

If the packets do not contain destination address A, they are associated with the PATTERN entity "dest_B".

The PATTERN entity "dest_B" contains a byte sequence representing destination address B. If the packets contain destination address B, they are assigned to the GROUP entity "interactive". If the packets do not contain destination address B, they are assigned to the GROUP entity "telnet_other".

The GROUP entity "interactive" is associated with class 1. This means that Telnet packets with destination address A or B have a priority of 1. The GROUP entity "telnet_other" is associated with class 3. This means that any other Telnet packets have a priority of 3.

7.12 Tasks Required to Set Up Pattern Matching

To set up pattern matching, you carry out the same tasks required to set up basic prioritization, except you need to:

❶ Set the match action of a packet to point to a PATTERN entity rather than a GROUP entity

❷ Create PATTERN entities

❸ Set the match action of PATTERN entities

❹ Set the mismatch action of PATTERN entities

Section 7.12.2 to Section 7.12.4 describe the four new tasks. The basic tasks are described in earlier sections of this chapter. The following list summarizes all the tasks, and the sections in which they are described:

1. Create the PRIORITY module (Section 7.7.1)
2. Create PACKET entities (Section 7.7.2)
3. Create GROUP entities (Section 7.7.3)

4. Set the match action of the PACKET entity to point to a PATTERN

   This new task is described in Section 7.12.1.

5. Create PATTERN entities

   This new task is described in Section 7.12.2.

6. Set the match action of each PATTERN entity

   This new task is described in Section 7.12.3.

7. Set the mismatch action of each PATTERN entity

   This new task is described in Section 7.12.4.

8. Assign a class to each GROUP entity (Section 7.7.5)

9. Set the DEFAULT CLASS associations (Section 7.7.6)

10. Create INTERFACE entities (Section 7.7.7)

11. Associate classes with queues (Section 7.7.8)

12. Enable the INTERFACE entities (Section 7.7.9)

13. Enable the PRIORITY module (Section 7.7.10)

If you want to add pattern matching to an existing prioritization set up, you should:

- Uncouple the PRIORITY module
- Complete the four new tasks
- Update the PRIORITY module

Note: The PRIORITY module must be Uncoupled before you can create or set any PATTERN entities.

7.12.1 Set the Match Action of PACKET Entities

To set the match action of a PACKET entity to point to a PATTERN entity, enter the following command:

```
NCL> SET PRIORITY PACKET packet-name -
_NCL> MATCH ACTION = PRIORITY PATTERN pattern-name
```

where: *packet-name* is the name of a PACKET entity and *pattern-name* is the name of a PATTERN entity.

Example

```
NCL> SET PRIORITY PACKET telnet MATCH ACTION = PRIORITY PATTERN dest_A
```

7.12.2 Create PATTERN Entities

To create a PATTERN entity, enter the following command:

```
NCL> CREATE PRIORITY PATTERN pattern-name FROM = count-offset-from , -
_NCL> OFFSET = n , STRING = %Xbyte-sequence , MASK = %Xbyte-mask
```

where:

pattern-name is the name of the pattern. It is useful to choose a name that represents the byte sequence described in the pattern

count-offset-from is either START or NEXT ISO LAYER

n is a decimal number

byte-sequence is a string of hex digits

byte-mask is a string of hex digits

Example

```
NCL> CREATE PRIORITY PATTERN dest_A FROM = start , -
_NCL>  OFFSET = 16 , STRING = %X1024101C , MASK = %XFFFFFFFF
```

7.12.3 Set the MATCH ACTION of Each PATTERN Entity

The match action of a PATTERN entity can either point to another PATTERN entity or a GROUP entity.

To set the match action to point to a PATTERN entity, enter the following command:

```
NCL> SET PRIORITY PATTERN pattern-name MATCH ACTION = PRIORITY -
_NCL> PATTERN pattern-name
```

To set the match action to point to a GROUP entity, enter the following command:

```
NCL> SET PRIORITY PATTERN pattern-name MATCH ACTION = PRIORITY -
_NCL> GROUP group-name
```

Example

```
NCL> SET PRIORITY PATTERN dest_A MATCH ACTION = PRIORITY GROUP interactive
```

7.12.4 Set the MISMATCH ACTION of Each PATTERN Entity

The mismatch action of a PATTERN entity can either point to another PATTERN entity or a GROUP entity.

To set the mismatch action to point to a PATTERN entity, enter the following command:

```
NCL> SET PRIORITY PATTERN pattern-name MISMATCH ACTION = PRIORITY -
_NCL> PATTERN pattern-name
```

To set the mismatch action to point to a GROUP entity, enter the following command:

```
NCL> SET PRIORITY PATTERN pattern-name MISMATCH ACTION = PRIORITY GROUP -
_NCL> group-name
```

Example

```
NCL> SET PRIORITY PATTERN dest_A MISMATCH ACTION = PRIORITY PATTERN dest_B
```

7.12.5 Examples of Setting Up Pattern Matching

This section shows the NCL commands required to set up the pattern matching shown in Figure 7–13 and Figure 7–14. These NCL commands are included in example script files. From the example script files, you can copy the NCL commands you require into the user NCL script files for your DECNIS. For more details about the user NCL script files, refer to Section 1.5.6.

For details about the location of the example script files on various load hosts, refer to Appendix G.

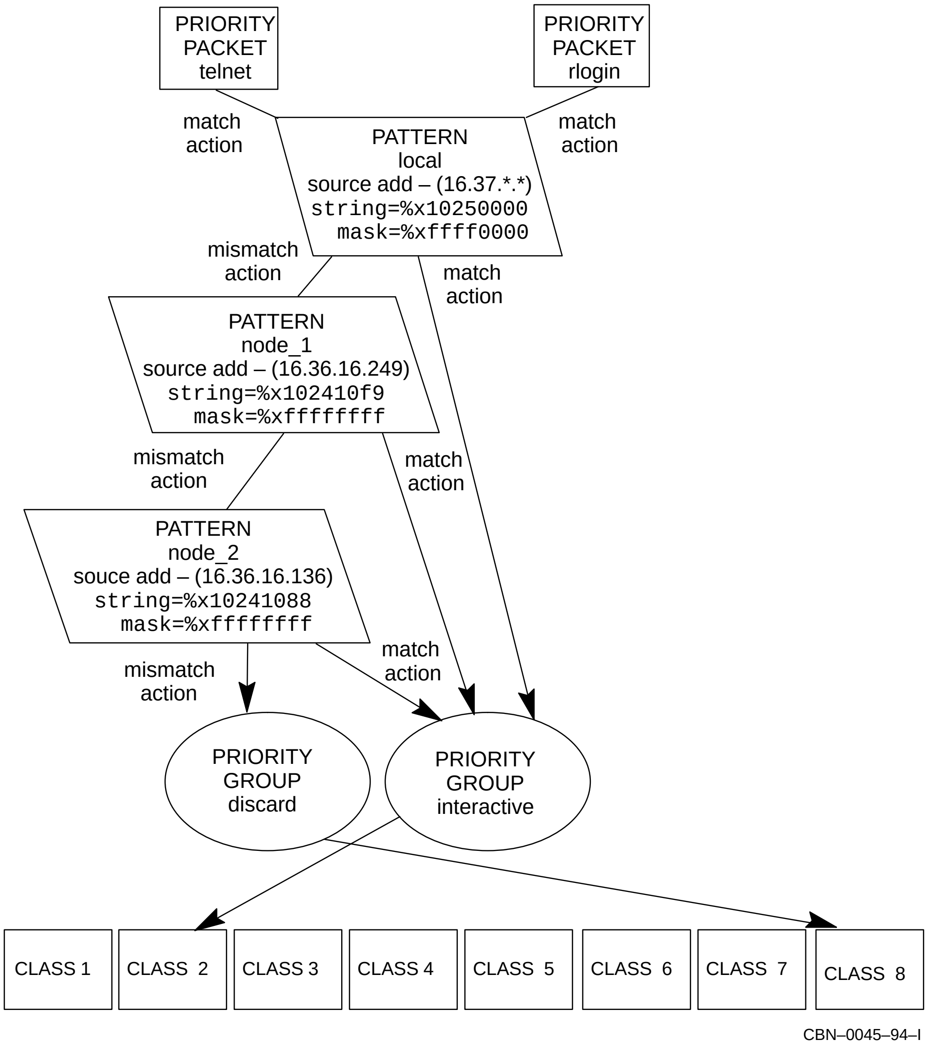
Example One

In this example, pattern matching is used to give a high priority to Telnet and RLOGIN sessions from all local nodes (16.37.*.*), and two remote nodes (16.36.16.249, 16.36.16.136). All other Telnet and RLOGIN sessions are discarded.

```
NCL> CREATE PRIORITY

NCL> CREATE PRIORITY PACKET telnet TYPE ip tcp, ip port range = [23..23]
NCL> CREATE PRIORITY PACKET rlogin TYPE ip tcp, ip port range = [513..513]

NCL> CREATE PRIORITY GROUP interactive
NCL> CREATE PRIORITY GROUP discard

NCL> SET PRIORITY PACKET telnet MATCH ACTION = PRIORITY PATTERN local
NCL> SET PRIORITY PACKET rlogin MATCH ACTION = PRIORITY PATTERN local

NCL> CREATE PRIORITY PATTERN local FROM = start, -
_NCL> OFFSET = 12, STRING = %X10250000 , MASK = %Xffff0000

NCL> CREATE PRIORITY PATTERN node_1 FROM = start, -
_NCL> OFFSET = 12, STRING = %X102410f9 , MASK = %Xffffffff

NCL> CREATE PRIORITY PATTERN node_2 FROM = start, -
_NCL> OFFSET = 12, STRING = %X10241088 , MASK = %Xffffffff

NCL> SET PRIORITY PATTERN local MATCH ACTION = PRIORITY GROUP interactive
NCL> SET PRIORITY PATTERN local MISMATCH ACTION = PRIORITY PATTERN node_1

NCL> SET PRIORITY PATTERN node_1 MATCH ACTION = PRIORITY GROUP interactive
NCL> SET PRIORITY PATTERN node_1 MISMATCH ACTION = PRIORITY PATTERN node_2

NCL> SET PRIORITY PATTERN node_2 MATCH ACTION = PRIORITY GROUP interactive
NCL> SET PRIORITY PATTERN node_2 MISMATCH ACTION = PRIORITY GROUP discard

NCL> SET PRIORITY GROUP interactive ASSIGNED CLASS 2
NCL> SET PRIORITY GROUP discard ASSIGNED CLASS 8

NCL> CREATE PRIORITY INTERFACE w622-5-0 COMMUNICATION PORT w622-5-0

NCL> SET PRIORITY INTERFACE w622-5-0 QUEUE 8 MAXIMUM QUEUE LENGTH = 0

NCL> ENABLE PRIORITY INTERFACE w622-5-0
NCL> ENABLE PRIORITY
```

The PATTERN entity "local" checks all Telnet and RLOGIN sessions to determine if they are from a local node. If they are local, they are assigned to GROUP entity "interactive".

Figure 7–13 Example One



If they are not local, they are checked against PATTERN entities "node_1" and "node_2" to determine if they are from an allowed remote node. If they are from an allowed remote node, they are assigned to the GROUP entity "interactive".

If they are not from an allowed node, they are assigned to the GROUP entity "discard".

The maximum queue length of queue 8 is set to zero. This ensures that all packets assigned to this queue are discarded.

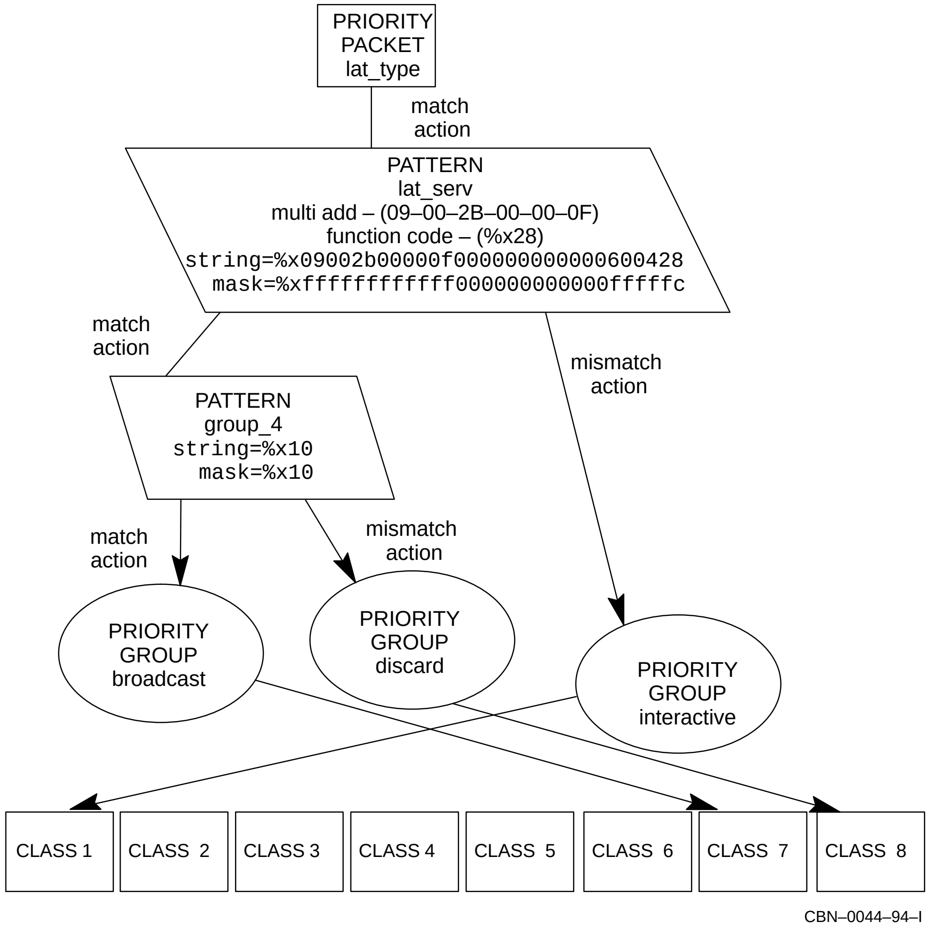
Example Two

In this example, pattern matching is used to discard all LAT service advertisements apart from group 4 advertisements (which have a priority of 7). All other LAT packets have a priority of 1.

```
NCL> CREATE PRIORITY

NCL> CREATE PRIORITY PACKET lat_type TYPE bridge type , -
_NCL> bridge ethernet protocol type = 60-04

NCL> CREATE PRIORITY GROUP interactive
NCL> CREATE PRIORITY GROUP broadcast
NCL> CREATE PRIORITY GROUP discard

NCL> SET PRIORITY PACKET lat_type MATCH ACTION = PRIORITY PATTERN lat_serv

NCL> CREATE PRIORITY PATTERN lat_serv FROM = start, -
_NCL> OFFSET = 0, STRING = %X09002b00000f000000000000600428 , -
_NCL> MASK = %Xffffffffffff000000000000fffffc

NCL> CREATE PRIORITY PATTERN group_4 FROM = next iso layer, -
_NCL> OFFSET = 13, STRING = %X10 , MASK = %X10

NCL> SET PRIORITY PATTERN lat_serv MATCH ACTION = PRIORITY PATTERN group_4
NCL> SET PRIORITY PATTERN lat_serv -
_NCL> MISMATCH ACTION = PRIORITY GROUP interactive

NCL> SET PRIORITY PATTERN group_4 MATCH ACTION = PRIORITY GROUP broadcast
NCL> SET PRIORITY PATTERN group_4 MISMATCH ACTION = PRIORITY GROUP discard

NCL> SET PRIORITY GROUP interactive ASSIGNED CLASS 1
NCL> SET PRIORITY GROUP broadcast ASSIGNED CLASS 7
NCL> SET PRIORITY GROUP discard ASSIGNED CLASS 8

NCL> CREATE PRIORITY INTERFACE w622-5-0 COMMUNICATION PORT w622-5-0

NCL> SET PRIORITY INTERFACE w622-5-0 QUEUE 8 MAXIMUM QUEUE LENGTH = 0

NCL> ENABLE PRIORITY INTERFACE w622-5-0
NCL> ENABLE PRIORITY
```

The PATTERN entity "lat_serv" checks all LAT packets to determine if they are LAT service advertisements. If they are not LAT service advertisements, they are assigned to the GROUP entity "interactive".

Figure 7–14 Example Two



If they are LAT service advertisements, they are checked against the PATTERN entity "group_4" to determine if they are group 4 advertisements. If they are group 4 advertisements, they are assigned to the GROUP entity "broadcast".

If they are not group 4 advertisements, they are assigned to the GROUP entity "discard".

The maximum queue length of queue 8 is set to zero. This ensures that all packets assigned to this queue are discarded.

8 Connections to the Network

8.1 Introduction

This chapter describes how to create and delete data link connections between the DECNIS and the network.

8.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

8.2 Use of Network Connections by the DECNIS

Generally, you can use the same connection for a number of different functions. For example, an HDLC data link can be used by a routing circuit, a MOP circuit and a remote bridge port; a CSMA/CD data link can be used by any combination of routing, bridging, and LLC2 functions.

8.3 Port Naming Scheme

8.3.1 Port Name Format

Each hardware port (sometimes called a communication port) on the DECNIS is identified by a name in the following format:

card-slot-port

card identifies the type of Network Interface Card; for example, W622 identifies the DEC WANcontroller 622 Network Interface Card.

slot is the number of the slot on the DECNIS backplane into which the Network Interface Card is inserted. Slot 1 is normally occupied by the management processor card and slot 2 by the pool memory board. The remaining slots are for Network Interface Cards.

port is a number identifying the port on the Card.

8.3.2 Examples

Table 8–1 shows some example port names.

Table 8–1 Example Port Names

Card	Slot	Port	Port Name
LANcontroller 601	3	0	L601-3-0
WANcontroller 622	4	0	W622-4-0
WANcontroller 622	4	1	W622-4-1
FDDIcontroller 621 on DECNIS 600	5 and 6	0	F621-6-0
ATMcontroller 631 on DECNIS 600	7 and 8	0	W631-8-0

8.4 Adding a Network Interface Card

8.4.1 Inserting the Network Interface Card

Insert the Network Interface Card into the hardware unit following the instructions in the *Installation and Service Manual* for your hardware.

You can add a Network Interface Card into a previously unused slot without powering down the DECNIS: see the *Installation and Service Manual* for your hardware for details of the procedure. Section 8.5 explains live insertion in more detail.

8.4.2 Configuring a New Network Interface Card

Follow these steps to configure a Network Interface Card.

Note that you can configure a Network Interface Card without it being present in the hardware slot. The Network Interface Card will then be available for use when it is inserted.

1. Create a DEVICE UNIT entity for the Network Interface Card:

   ```
   NCL> CREATE DEVICE UNIT unit-name NAME device
   ```

where:

unit-name is a name to identify the DEVICE UNIT entity. For simplicity, you are recommended to make this the same as *device*.

device identifies the type of Network Interface Card, the slot in which it is positioned, and, for certain Network Interface Card types, the port (see Section 8.3).

Enter one of the following values for *device*:

L601-*X*
L602-*X*-0
L602-*X*-1
F621-*X*
W618-*X*
W614-*X*
W622-*X*-0
W622-*X*-1
W631-*X*

X is the slot in which the Network Interface Card is positioned.

F621 and W631 Network Interface Cards: Note that each of these cards occupy two slots.

For the DECNIS 600, use the higher-numbered slot.

For the DECNIS 500, use the lower-numbered slot for the F621 Interface Card. As the W631 Network Interface Card occupies two slots and does not support MOP loading, it cannot be used in the DECNIS 500.

2. Enable the device unit:

```
NCL> ENABLE DEVICE UNIT unit-name
```

3. To ensure that the Network Interface Card has loaded successfully, enter the following command after the card has been inserted and has successfully completed its self-test:

```
NCL> SHOW DEVICE UNIT unit-name STATUS
```

The status returned should be Running.

8.4.3 W622 and L602 LED Displays

When you use the second line of a WANcontroller 622 or LANcontroller 602 Network Interface Card without using the first line, the green LED stays in fast-flashing mode. To avoid this, create and enable a DEVICE UNIT entity for the first line.

The DECNIS configurators (text-based and Windows) always create and enable device units for both lines, avoiding this problem.

8.5 Live Insertion

8.5.1 Definition

Live Insertion allows you to insert a Network Interface Card into a previously unused slot of the hardware unit, without powering down and rebooting the DECNIS.

8.5.2 Self-Test

Note that you must wait until the DECNIS initialization is complete before you insert a new Network Interface Card in this way. If you insert a card after the DECNIS has been powered up but before its CMIP initialization script has been executed, the card's self-test will not operate correctly.

8.5.3 Removing the Network Interface Card

Once the Network Interface Card's microcode has loaded, you cannot remove the card from the slot without crashing the DECNIS.

8.5.4 How a Card's Microcode Is Loaded

The microcode is loaded in either of the following circumstances:

- In the DECNIS configurator, you have specified the card in its correct slot on the Network Interface Cards screen. (In the clearVISN DECNIS configurator, click the **Interfaces** button to go to this screen.)
- The card is physically present in the hardware slot when the DECNIS is booted.
- You have enabled the DEVICE UNIT entity representing the card on a running system, and the card is physically present in the hardware slot.

8.5.5 Restriction: F621 and W631 Network Interface Cards

You cannot insert an F621 or W631 Network Interface Card into a previously unused slot. If you attempt to do this, the card will fail to load its microcode.

8.5.6 Examples

Typical uses of live insertion would be to:

- Maintain the functionality of a DECNIS if a Network Interface Card fails. For example, if a LANcontroller 601 card in slot 3 fails, a replacement LANcontroller 601 card can be positioned in an unused slot of the DECNIS and configured, without rebooting. You should not remove the failed card until the DECNIS is next powered down.
- Upgrade the DECNIS with extra network connections, without rebooting.

8.6 Creating an ATM Connection

8.6.1 Introduction

This section describes how to configure an ATM connection on a suitable hardware port. It also describes how to configure an ATM connection with FLOWmaster™ flow control.

Once you have set up an ATM connection, you need to do the following:

- To route over ATM, create ATM Permanent routing circuits, as described in Section 9.6.
- To bridge over ATM, create bridge ports, as described in Section 21.5.3.

8.6.1.1 ATM Modules

In order to configure an ATM connection, you need to set up three modules:

- The Multiplexed Interface module is a physical layer entity that defines and manages the characteristics of the physical line used for ATM on the DECNIS.
- The ATM Connection Management module is a physical layer entity that defines and manages access to the available bandwith for ATM Permanent Virtual Circuits (ATM PVCs).
- The ATM Multiprotocol Encapsulation module is a data link layer entity that provides a PPP-like WAN service. This service allows multi-protocol routing and bridging over ATM.

Figure 9–4 shows the relation between these modules, and how they relate to an ATM Permanent routing circuit. Figure 21–4 shows how the ATM modules relate to an ATM Permanent bridge port.

8.6.1.2 Requirement

You can only set up ATM connections on the ATMcontroller 631 Network Interface Card.

8.6.2 Procedure

Follow these steps to create a new ATM connection:

1. Ensure that the Multiplexed Interface module exists by entering the following command:

   ```
   NCL> CREATE MULTIPLEXED INTERFACE
   ```

   Result: A message will tell you if the Multiplexed Interface module already exists; otherwise, it will be created.

2. Ensure that the ATM Connection Management module exists by entering the following command:

   ```
   NCL> CREATE ATM CONNECTION MANAGEMENT
   ```

   Result: A message will tell you if the ATM Connection Management module already exists; otherwise, it will be created.

3. Ensure that the ATM Multiprotocol Encapsulation module exists by entering the following command:

   ```
   NCL> CREATE ATM MULTIPROTOCOL ENCAPSULATION
   ```

   Result: A message will tell you if the ATM Multiprotocol Encapsulation module already exists; otherwise it will be created.

4. Create the multiplexed line and specify the communications port it will use by entering the following command:

   ```
   NCL> CREATE MULTIPLEXED INTERFACE LINE line-name -
   _NCL> COMMUNICATIONS PORT W631-x-0, -
   _NCL> INTERFACE TYPE interface-type, -
   _NCL> FRAMING TYPE framing-type
   ```

   where:

   line-name is the name of the line.

   x is the number of the hardware slot in which the Network Interface Card is positioned.

   interface-type identifies the interface type of the physical line and is one of DS3, E3, or OC-3

   framing-type identifies the framing type of the physical line. This must be compatible with the interface type. Valid options are:

   - DS3-CBitParity or DS3-ClearChannel for DS3.
   - E3-G832 or E3-G751 for E3.
   - SONET or SDH for OC-3.

   If no framing type is specified, a default is used based on the interface type.

   Result: Either the multiplexed interface line will be created, or a message will tell that the combinations you have selected are incompatible.

5. Enter the following command to set the clock timing source and line equalization build-out setting for the physical line:

   ```
   NCL> SET MULTIPLEXED INTERFACE LINE line-name -
   _NCL> CLOCK SOURCE clock-source, -
   _NCL> LINE LENGTH line-length, -
   ```

where:

clock-source specifies whether the clock timing is derived from the network or from a local source. The default is Network.

line-length is the length of the RG59B cable attached to the device. For DS3 or E3, and for cable lengths above 125 feet, it is important to set the line length to ensure that the signal is boosted appropriately. Note that the actual line length must not exceed 450 feet.

6. Create the logical channel to be used for this line. Note that the value {0} is mandatory for the MAP attribute. This value indicates that the logical channel is occupying the entire usable line bandwidth; this is known as Clear Channel mode. The DECNIS does not support channelized operation.

   Enter the following command:

   ```
   NCL> CREATE MULTIPLEXED INTERFACE LINE line-name -
   _NCL> LOGICAL CHANNEL channel-name MAP {0}
   ```

   where *channel-name* is the multiplexed interface logical channel name.

   Result: The logical channel is created.

7. Create the ATM Connection Management Line entity for the connection. If you do **not** wish to enable FLOWmaster flow control on the Line, enter the following command:

   ```
   NCL> CREATE ATM CONNECTION MANAGEMENT LINE line-name
   ```

   If you wish to enable FLOWmaster flow control on the Line, refer to Section 8.6.3.

8. Associate the Multiplexed Interface Logical Channel with the ATM Connection Management line. Enable or disable PLCP and Cell Scrambling.

   ```
   NCL> SET ATM CONNECTION MANAGEMENT line-name -
   _NCL> LOWER LAYER ENTITY MULTIPLEXED INTERFACE LINE line-name -
   _NCL> LOGICAL CHANNEL channel-name, -
   _NCL> PLCP plcp, -
   _NCL> CELL SCRAMBLING cell-scrambling
   ```

   where:

   plcp is either enabled or disabled.

   cell-scrambling is either enabled or disabled.

   Digital recommends that you always set cell scrambling to Enabled, and that you set PLCP to Disabled for connection to an ATM switch. However, note also that the values you set must match those of the target system.

9. Create the ATM Connection Management Permanent Virtual Circuit (PVC) and associate it with the Virtual Circuit Identifier (VCI). If you do **not** wish to enable FLOWmaster flow control on the PVC, enter the following command:

   ```
   NCL> CREATE ATM CONNECTION MANAGEMENT LINE line-name PVC pvc-name -
   _NCL> VIRTUAL CIRCUIT IDENTIFIER vci-value, -
   ```

   where:

pvc-name	is the Permanent Virtual Circuit name.
vci-value	is the Virtual Circuit Identifier, in the range 32 through 1023.

   If you wish to enable FLOWmaster flow control on the PVC, refer to Section 8.6.3.

   Note that you may only create up to 100 ATM PVCs on the DECNIS.

10. Configure the circuit to match the capabilities of the switch and the configuration of the remote system:

    ```
    NCL> SET ATM CONNECTION MANAGEMENT LINE line-name PVC pvc-name -
    _NCL> PEAK RATE peak-rate, -
    _NCL> MINIMUM GUARANTEED RATE min-rate, -
    _NCL> TRAFFIC SHAPING PRIORITY shaping-priority
    ```

    where:

peak-rate	is the maximum rate at which the PVC is allowed to transmit, where the rate is a percentage of the bandwidth. Range: 1 to 100. Default: 100 If a PVC is given a peak rate of 100, that PVC may be able to use all of the available bandwidth. However, available bandwidth depends on the number and activity of PVCs, and the priority assigned to them.
min-rate	is the percentage of bandwidth guaranteed for the PVC. Range: 1 to 100. The default is 1.
shaping-priority	is the PVC priority, in the range 0 to 11. PVCs given a priority of 0 have top priority. The default is 0

    If you use the defaults, the bandwidth will be allocated evenly between all PVCs. It is recommended that you use the default values for T3 or E3 leased-line connections or when using a single PVC.

11. Create the ATM Multiprotocol Encapsulation link:

    ```
    NCL> CREATE ATM MULTIPROTOCOL ENCAPSULATION LINK link-name
    ```

12. Associate the ATM Multiprotocol Encapsulation link with the ATM Connection Management PVC:

```
NCL> SET ATM MULTIPROTOCOL ENCAPSULATION LINK link-name -
_NCL> LOWER LAYER ENTITY ATM CONNECTION MANAGEMENT LINE line-name -
_NCL> PVC pvc-name
```

13. Set the maximum Protocol Data Unit size of the ATM Multiprotocol Encapsulation link:

```
NCL> SET ATM MULTIPROTOCOL ENCAPSULATION LINK link-name
```

14. Enable the entities:

```
NCL> ENABLE MULTIPLEXED INTERFACE LINE line-name -
 _NCL> LOGICAL CHANNEL channel-name
NCL> ENABLE MULTIPLEXED INTERFACE LINE line-name
NCL> ENABLE ATM CONNECTION MANAGEMENT LINE line-name
NCL> ENABLE ATM MULTIPROTOCOL ENCAPSULATION LINK link-name
```

8.6.3 Setting Up FLOWmaster Flow Control

The ATM Forum has defined a class of service with the name Available Bit Rate (ABR), which provides LAN-like performance on ATM data links. This type of performance has the following properties:

- All of the unused link bandwidth is available to all the active ABR virtual circuits (VCs).
- The link bandwidth is shared fairly between those VCs.
- There is negligible cell loss.

The DECNIS FLOWmaster facility provides flow control of ABR traffic. FLOWmaster provides credit based flow control, per hop, and per virtual circuit (VC). As well as delivering high performance, FLOWmaster ensures that no cells are lost because of congestion in the ATM switching network; this can help avoid throughput collapse. It also enables flow-controlled VCs to use the full link bandwidth when the link becomes idle.

8.6.3.1 Procedure for Enabling FLOWmaster

You need to configure FLOWmaster on each PVC that will use ABR flow control. Follow these steps:

1. Create an ATM Connection Management Line with FLOWmaster enabled:

```
NCL> CREATE ATM CONNECTION MANAGEMENT LINE line-name -
_NCL> FLOWMASTER ENABLED
```

2. On the FLOWmaster line, create an ATM PVC, specifying the type of flow control that you want:

```
NCL> CREATE ATM CONNECTION MANAGEMENT LINE line-name PVC pvc-name -
_NCL> VIRTUAL CIRCUIT IDENTIFIER vci-value CIRCUIT TYPE type
```

where:

pvc-name is the Permanent Virtual Circuit name.

vci-value is the Virtual Circuit Identifier, in the range 32 through 1023.

type is one of the following:

CBR Constant Bit Rate. Traffic is scheduled according to the circuit's Minimum Guaranteed Rate, Peak Rate and Priority. Traffic is **not** flow controlled. This is the default.

ABR Available Bit Rate. Traffic is flow controlled by FLOWmaster. Circuits are scheduled in a round robin fashion. Peak Rate and Minimum Guaranteed Rate have no effect; that is, the circuit can use all the available bandwidth.

UBR Unconstrained Bit Rate. Traffic is **not** flow controlled by FLOWmaster. Circuits are scheduled in a round robin fashion. Peak Rate and Minimum Guaranteed Rate have no effect, as with ABR.

If you want FLOWmaster flow control, enter ABR as the CIRCUIT TYPE.

8.6.3.2 Establishing a FLOWmaster Link

When the ATM link first becomes available, the two end points run an initialization sequence to determine if both ends can support FLOWmaster operation. Flow control is only enabled if both end points support FLOWmaster.

If one end of the link does not support FLOWmaster, or the remote end of the link does not respond to the initialization messages, then any ABR circuits which have been configured will run without flow control, that is, they will act as UBR circuits.

8.6.4 Routing Over ATM Connections

See Section 9.6.

8.6.5 Bridging Over ATM Connections

See Section 21.5.

8.7 Creating a CSMA/CD Connection

8.7.1 Introduction

This section describes how to configure a CSMA/CD data link on a suitable hardware port.

Requirement

You can only set up CSMA/CD connections using a LAN Network Interface Card (for example, LANcontroller 601).

8.7.2 Procedure

Follow these steps to create a new CSMA/CD connection:

1. Ensure that the CSMA-CD module exists by entering the following command:

   ```
   NCL> CREATE CSMA-CD
   ```

   Result: A message will tell you if the CSMA-CD module already exists; otherwise, it will be created.

2. Create a CSMA-CD STATION entity for the connection, and specify the hardware port on which the connection will exist.

   ```
   NCL> CREATE CSMA-CD STATION station-name COMMUNICATION PORT port-name
   ```

   where:

   station-name is a name to identify the CSMA-CD station. Each CSMA-CD station on the DECNIS must have a unique name.

   port-name is the name of the hardware port (see Section 8.3).

3. Enable the CSMA-CD STATION entity:

   ```
   NCL> ENABLE CSMA-CD STATION station-name
   ```

4. Set up a MOP circuit on the data link, to allow subsequent loopback testing: see Section 8.26.

8.8 Creating an FDDI Connection

8.8.1 Introduction

This section describes how to configure an FDDI data link on a suitable hardware port.

Requirement

You can only set up FDDI connections using the FDDIcontroller 621 Network Interface Card.

8.8.2 Procedure

Follow these steps to create a new FDDI connection:

1. Ensure that the FDDI module exists by entering the following command:

   ```
   NCL> CREATE FDDI
   ```

   Result: A message will tell you if the FDDI module already exists; otherwise, it will be created.

2. Create an FDDI STATION entity for the connection, and specify the hardware port on which the connection will exist.

   ```
   NCL> CREATE FDDI STATION station-name COMMUNICATION PORT port-name
   ```

   where:

   station-name is a name to identify the FDDI station. Each FDDI station on the DECNIS must have a unique name.

   port-name is the name of the hardware port (see Section 8.3).

   Result: Creating an FDDI station entity automatically creates two child entities for the station: LINK and PHY PORT.

3. Enable the FDDI STATION entity:

   ```
   NCL> ENABLE FDDI STATION station-name
   ```

   Result: Enabling an FDDI station entity automatically enables the LINK and PHY PORT entities that are children of that station.

4. Set up a MOP circuit on the data link, to allow subsequent loopback testing: see Section 8.26.

8.8.3 Enabling the Ring Purger on an FDDI Connection

When you create an FDDI STATION LINK manually, the ring purger is turned on automatically.

However, if you create an FDDI connection using either of the DECNIS configurators, the ring purger set to off.

You can turn the ring purger on as follows:

```
NCL>DISABLE FDDI STATION station-name LINK n
NCL> SET FDDI STATION station-name LINK n -
_NCL> RING PURGER ENABLE TRUE

NCL> ENABLE FDDI STATION station-name LINK n
```

8.9 Creating a VCP Connection

8.9.1 Introduction

This section describes how to configure a VCP (Vitalink® Communications Protocol) data link on a suitable hardware port. VCP is used for synchronous connections between the DECNIS and a Vitalink TransLAN® bridge, using either a routing circuit or a bridge port.

Requirement

You can only set up VCP connections on a WANcontroller 622 Network Interface Card.

8.9.2 Procedure

A VCP link uses a variant of the CSMA-CD station running over a Modem Connect Line to transfer data. Follow these steps to create a new VCP connection:

1. Ensure that the MODEM CONNECT module exists by entering the following command:

   ```
   NCL> CREATE MODEM CONNECT
   ```

   Result: A message will tell you if the MODEM CONNECT module already exists; otherwise, it will be created.

2. Create a MODEM CONNECT LINE entity for the connection, and specify the hardware port on which the connection will exist.

   ```
   NCL> CREATE MODEM CONNECT LINE line-name COMMUNICATION PORT W622-slot-port
   ```

   where:

line-name	is the name of the line.
W622-*slot-port*	is the name of the hardware port (see Section 8.3).

3. Enable the MODEM CONNECT LINE entity:

   ```
   NCL> ENABLE MODEM CONNECT LINE line-name
   ```

   Result: You have now established a synchronous line.

4. Ensure that the CSMA-CD module exists by entering the following command:

   ```
   NCL> CREATE CSMA-CD
   ```

   Result: A message will tell you if the CSMA-CD module already exists; otherwise, it will be created.

5. Create a CSMA-CD STATION entity for the connection, and specify the station type as VIRTUAL VCP:

   ```
   NCL> CREATE CSMA-CD STATION station-name STATION TYPE VIRTUAL VCP
   ```

   where *station-name* is a name to identify the CSMA-CD station. Each CSMA-CD station on the DECNIS must have a unique name.

6. Associate the line created earlier with this data link:

   ```
   NCL> SET CSMA-CD STATION station-name LOWER LAYER -
   _NCL> ENTITY MODEM CONNECT LINE line-name
   ```

7. If the Vitalink end of the connection is a Turbo card, set the interframe gap to 2:

   ```
   NCL> SET CSMA-CD STATION station-name INTERFRAME GAP 2
   ```

   If the Vitalink end of the connection is not a Turbo card, leave the interframe gap at its default of 26.

8. Enable the CSMA-CD STATION entity:

   ```
   NCL> ENABLE CSMA-CD STATION station-name
   ```

9. Set up a MOP circuit on the data link, to allow subsequent loopback testing: see Section 8.26.

8.9.3 Routing Over VCP Data Links

The procedure for setting up a routing circuit over a VCP data link is exactly the same as for a physical CSMA/CD station: see Section 9.3.

8.9.4 Bridging Over VCP Data Links

Section 21.5 describes how to set up a bridge port on a VCP data link.

8.10 Creating an HDLC Connection

8.10.1 Introduction

This section describes how to configure an HDLC data link on a suitable synchronous hardware port.

Requirement

You can only set up HDLC connections on one of the following Network Interface Cards:

- WANcontroller 614
- WANcontroller 618

- WANcontroller 622

8.10.2 Procedure

Follow these steps to create a new HDLC connection:

1. Ensure that the MODEM CONNECT module exists by entering the following command:

   ```
   NCL> CREATE MODEM CONNECT
   ```

   Result: A message will tell you if the MODEM CONNECT module already exists; otherwise, it will be created.

2. Create a MODEM CONNECT LINE entity for the connection, specifying the hardware port on which the connection will exist, and the profile to be used for the line.

   ```
   NCL> CREATE MODEM CONNECT LINE line-name -
   _NCL> COMMUNICATION PORT port-name, PROFILE profile-name
   ```

   where:

   profile-name is either "NORMAL" or "DATEXP". The profile specifies the default value and permitted range for certain timers on the line. See the NCL online help for details of these timers.

   port-name is the name of the hardware port (see Section 8.3).

3. Set the MODEM CONTROL characteristic for the line:

   ```
   NCL> SET MODEM CONNECT LINE line-name MODEM CONTROL control
   ```

   where *control* is either FULL or NONE. FULL is the normal setting, and means that the line takes note of the modem control signals. Set it to NONE if the line is to ignore all modem control signals (for example, during loopback testing).

4. Enable the MODEM CONNECT LINE entity:

   ```
   NCL> ENABLE MODEM CONNECT LINE line-name
   ```

   Result: You have now established a synchronous line.

5. Ensure that the HDLC module exists:

   ```
   NCL> CREATE HDLC
   ```

6. Create an HDLC data link for the circuit, and specify its type.

   ```
   NCL> CREATE HDLC LINK link-name LINKTYPE BALANCED
   ```

   where *link-name* is a name identifying the data link. You should make this the same as the communication port name.

7. Associate the LINE created earlier with this data link:

   ```
   NCL> SET HDLC LINK link-name -
   _NCL> PHYSICAL LINE MODEM CONNECT LINE line-name
   ```

8. Enable the data link:

   ```
   NCL> ENABLE HDLC LINK link-name
   ```

9. Create and enable a logical station for the link:

   ```
   NCL> CREATE HDLC LINK link-name LOGICAL STATION station-name
   ```

   ```
   NCL> ENABLE HDLC LINK link-name LOGICAL STATION station-name
   ```

   For simplicity, use the same name for the logical station as for the link.

10. Set up a MOP circuit on the data link, to allow subsequent loopback testing: see Section 8.26.

8.11 Changing the Acknowledge and Holdback Timers on an HDLC Connection

8.11.1 Function of the Acknowledge Timer

After sending a message, a data link will wait for a time determined by the acknowledge timer. If the data link does not receive a response within this time, it will retransmit the message.

8.11.2 Function of the Holdback Timer

The holdback timer specifies the delay (in milliseconds) before an explicit acknowledgment must be sent if there are no data frames available to carry an implicit acknowledgment.

8.11.3 Guidelines for Setting Timers

Decrease the value of the acknowledge timer from the default value (3000) if you are using a fast or noisy link. Increase this value if you are using a slow link.

The values of the acknowledge timer and holdback timer should be related as follows:

```
Acknowledge Timer >= (2 * maxfrm) + Holdback Timer
```

where `maxfrm` is the time taken for a maximum-sized frame to be transmitted by the DECNIS and received at the adjacent system, and `Holdback Timer\` is the value of the holdback timer at the adjacent system.

8.11.4 Setting the Acknowledge Timer

Set the ACKNOWLEDGE TIMER characteristic of the data link:

```
NCL> SET HDLC LINK link-name ACKNOWLEDGE TIMER n
```

where *n* is a decimal number between 1 and 60,000, and specifies the time in milliseconds.

Changing a Running System

Note that when you change the value of this timer on a running system, the timer is reset.

8.11.5 Setting the Holdback Timer

Set the HOLDBACK TIMER characteristic of the data link:

```
NCL> SET HDLC LINK link-name HOLDBACK TIMER n
```

where *n* is a decimal number between 0 and 60,000, and specifies the time in milliseconds.

Changing a Running System

Note that when you change the value of this timer on a running system, the timer is reset.

8.12 Changing Preferred Window Size on an HDLC Connection

8.12.1 Function of Preferred Window Size

The value of the preferred window size specifies the number of frames that can be sent on an HDLC data link without waiting for an acknowledgment.

8.12.2 Guideline for Setting Preferred Window Size

Increase the value from the default (2) if there is a significant delay on the link.

8.12.3 Setting the Preferred Window Size

Set the PREFERRED WINDOW SIZE characteristic of the data link:

```
NCL> SET HDLC LINK link-name PREFERRED WINDOW SIZE n
```

where *n* is a decimal number between 1 and 127.

If value is greater than 7

Note that if you set the PREFERRED WINDOW SIZE characteristic to a value greater than 7, you must set the SEQUENCE MODULUS characteristic of the link to 128.

8.13 HDLC Transmit Window Size

The W618 and W614 Network Interface Cards only have limited buffering and cannot operate multiple lines with very large HDLC transmit window sizes.

You can change the window size to be offered during negotiation by using the following command:

```
NCL> SET HDLC LINK link-name PREFERRED WINDOW SIZE n
```

The W618 and W614 Network Interface Cards will not make use of any window size greater than 64. Larger values may be set, but no line will ever have more than 64 frames waiting for acknowledgment.

8.14 HDLC and Frame Forwarding Services

If you are using HDLC links to a frame forwarding service, for example, a Stratacom IPX mux, then you must make sure that the link is compatible with the service. This applies particularly to the CRC type.

Set the CRC type as follows:

```
NCL> SET HDLC LINK link-name PREFERRED CRC TYPE type
```

where *type* is one of 16 BIT, 32 BIT, or EITHER.

8.15 Creating a DDCMP Connection

8.15.1 Introduction

This section describes how to configure a DDCMP data link on a suitable serial hardware port. You can set up synchronous and asynchronous links as suited to the modems and remote stations. If you have a choice, synchronous mode will give you better line performance.

Requirement

You can only set up DDCMP connections on the WANcontroller 614 or WANcontroller 618 Network Interface Cards.

8.15.2 Procedure

Follow these steps to create a new DDCMP connection:

1. Ensure that the MODEM CONNECT module exists by entering the following command:

   ```
   NCL> CREATE MODEM CONNECT
   ```

   Result: A message will tell you if the MODEM CONNECT module already exists; otherwise, it will be created.

2. Create a MODEM CONNECT LINE entity for the connection, specifying the hardware port on which the connection will exist, and the profile and mode to be used for the line.

   ```
   NCL> CREATE MODEM CONNECT LINE line-name -
   _NCL> COMMUNICATION PORT port-name, PROFILE profile-name,
   _NCL> COMMUNICATION MODE mode
   ```

   where:

   profile-name is either "NORMAL" or "DATEXP". The profile specifies the default value and permitted range for certain timers on the line.

   port-name is the name of the hardware port (see Section 8.3).

   mode is the mode used (either SYNCHRONOUS or ASYNCHRONOUS)

3. Set the MODEM CONTROL characteristic for the line:

   ```
   NCL> SET MODEM CONNECT LINE line-name MODEM CONTROL control
   ```

   where *control* is either FULL or NONE. FULL is the normal setting, and means that the line takes note of the modem control signals. Set it to NONE if the line is to ignore all modem control signals (for example, during loopback testing) or if the modem or connection does not use them.

   If you are setting up an asynchronous link, set the SPEED characteristic for the line:

   ```
   NCL> SET MODEM CONNECT LINE line-name SPEED n
   ```

   where *n* is the speed required. The default value is 9600.

4. Enable the MODEM CONNECT LINE entity:

   ```
   NCL> ENABLE MODEM CONNECT LINE line-name
   ```

   Result: You have now established a line.

5. Ensure that the DDCMP module exists:

   ```
   NCL> CREATE DDCMP
   ```

6. Create a DDCMP data link for the circuit, and specify its protocol type.

   ```
   NCL> CREATE DDCMP LINK link-name PROTOCOL POINT
   ```

   where *link-name* is a name identifying the data link. You should make this the same as the communication port name. The PROTOCOL is POINT for DDCMP links.

7. Associate the LINE created earlier with this data link:

   ```
   NCL> SET DDCMP LINK link-name -
   _NCL> PHYSICAL LINE MODEM CONNECT LINE line-name
   ```

8. Enable the data link:

   ```
   NCL> ENABLE DDCMP LINK link-name
   ```

9. Create and enable a logical station for the link:

   ```
   NCL> CREATE DDCMP LINK link-name LOGICAL STATION station-name
   NCL> ENABLE DDCMP LINK link-name LOGICAL STATION station-name
   ```

   For simplicity, use the same name for the logical station as for the link.

10. Set up a MOP circuit on the data link, to allow subsequent loopback testing: see Section 8.26.

8.16 Changing the Retransmit and Holdback Timers on a DDCMP Connection

8.16.1 Function of the Retransmit Timer

After sending a message, a data link will wait for a time determined by the retransmit timer. If the data link does not receive a response within this time, it will attempt recovery action.

8.16.2 Function of the Holdback Timer

The HOLDBACK TIMER characteristic specifies the delay (in milliseconds) before an explicit acknowledgment must be sent if there are no data frames available to carry an implicit acknowledgment.

8.16.3 Guidelines for Setting Timers

Decrease the value of the retransmit timer from the default value (3000) if you are using a fast or noisy link. Increase this value if you are using a slow link.

The values of the retransmit timer and holdback timer should be related as follows:

```
Retransmit Timer >= (2 * maxfrm) + Holdback Timer
```

where `maxfrm` is the time taken for a maximum-sized frame to be transmitted by the DECNIS and received at the adjacent system, and `Holdback Timer` is the value of the holdback timer at the adjacent system.

8.16.4 Setting the Retransmit Timer

Set the RETRANSMIT TIMER characteristic of the data link:

```
NCL> SET DDCMP LINK link-name RETRANSMIT TIMER n
```

where *n* is a decimal number between 1 and 60,000, and specifies the time in milliseconds.

Changing a Running System

Note that when you change the value of this timer on a running system, the timer is reset.

8.16.5 Setting the Holdback Timer

Set the HOLDBACK TIMER characteristic of the data link:

```
NCL> SET DDCMP LINK link-name LOGICAL STATION station-name HOLDBACK TIMER n
```

where *n* is a decimal number between 0 and 60,000, and specifies the time in milliseconds.

Changing a Running System

Note that when you change the value of this timer on a running system, the timer is reset.

8.17 Changing the Transmit Window on a DDCMP Connection

8.17.1 Function of Transmit Window

The value of the transmit window specifies the number of data messages that the local station can send without receiving an acknowledgment.

8.17.2 Guideline for Setting Transmit Window Size

The DECNIS text-based configurator and the clearVISN DECNIS configurator set the transmit window size at 255. Only change this value if the remote station has limited transient receive buffering. Set it to a large enough value so that the window does not close in the time taken for an acknowledgment to arrive.

8.17.3 Setting the Transmit Window Size

Set the TRANSMIT WINDOW characteristic of the data link:

```
NCL> SET DDCMP LINK link-name LOGICAL STATION station-name TRANSMIT WINDOW n
```

where *n* is a decimal number between 1 and 255.

8.18 Creating a PPP Connection

8.18.1 Introduction

This section describes how to configure a PPP data link on a suitable synchronous hardware port. If you want to set up a PPP connection over a frame relay network, follow the procedures in Section 8.20.

Requirement

You can only set up PPP connections using one of the following Network Interface Cards:

- WANcontroller 614
- WANcontroller 618
- WANcontroller 622

8.18.2 Procedure

Follow these steps to create a new PPP connection:

1. Ensure that the MODEM CONNECT module exists by entering the following command:

   ```
   NCL> CREATE MODEM CONNECT
   ```

   Result: A message will tell you if the MODEM CONNECT module already exists; otherwise, it will be created.

2. Create a MODEM CONNECT LINE entity for the connection, specifying the hardware port on which the connection will exist, and the profile to be used for the line:

```
NCL> CREATE MODEM CONNECT LINE line-name -
_NCL> COMMUNICATION PORT port-name, PROFILE profile-name
```

where:

profile-name is either "NORMAL" or "DATEXP". The profile specifies the default value and permitted range for certain timers on the line. See the NCL online help for details of these timers.

port-name is the name of the hardware port (see Section 8.3).

3. Set the MODEM CONTROL characteristic for the line:

   ```
   NCL> SET MODEM CONNECT LINE line-name MODEM CONTROL control
   ```

   where *control* is either FULL or NONE. FULL is the normal setting, and means that the line takes note of the modem control signals. Set it to NONE if the line is to ignore all modem control signals (for example, during loopback testing).

4. Enable the MODEM CONNECT LINE entity:

   ```
   NCL> ENABLE MODEM CONNECT LINE line-name
   ```

   Result: You have now established a synchronous line.

5. Ensure that the PPP module exists:

   ```
   NCL> CREATE PPP
   ```

6. Create a PPP data link for the circuit, and specify its type:

   ```
   NCL> CREATE PPP LINK link-name TYPE SYNCHRONOUS
   ```

   where *link-name* is a name identifying the data link. You should make this the same as the communication port name.

7. Associate the LINE created earlier with this data link:

   ```
   NCL> SET PPP LINK link-name -
   _NCL> LOWER LAYER ENTITY MODEM CONNECT LINE line-name
   ```

8. Enable the data link:

   ```
   NCL> ENABLE PPP LINK link-name
   ```

8.18.3 Creating a PPP Connection for NetWare IPX

If you wish to run native NetWare IPX routing on the PPP connection, follow these steps:

1. Carry out Steps 1 to 6 in Section 8.18.2.
2. Specify that the NetWare IPX control protocol be used on the PPP link.

   ```
   NCL> ADD PPP LINK link-name REQUIRED CONTROL PROTOCOL {NOVELL IPX}
   ```

3. If NetWare IPX is the only routing protocol to be run on this link, you may need to remove other routing protocols that were set up by default. First, find out what other protocols are set up for the link:

   ```
   NCL> SHOW PPP LINK link-name REQUIRED CONTROL PROTOCOL {}
   ```

4. Then, remove the protocols that you do not require. For example:

   ```
   NCL> REMOVE PPP LINK link-name REQUIRED CONTROL PROTOCOL {DECNET PHASE IV}
   ```

5. Enable the data link:

   ```
   NCL> ENABLE PPP LINK link-name
   ```

8.18.4 DECnet Phase IV Routing Over PPP: Interoperability with Other Vendors

If the DECNIS receives a PPP frame containing a DECnet Phase IV packet that does not entirely fill the data portion of the PPP frame (that is, there are padding octets between the end of the DECnet Phase IV packet and the end of the PPP frame), then the DECNIS will discard the packet.

The DECNIS always sends PPP frames containing a DECnet Phase IV packet that exactly fills the data portion of the PPP frame.

8.19 Creating a CHDLC Connection

8.19.1 Introduction

This section describes how to configure a CHDLC data link on a suitable synchronous hardware port. If you want to set up a CHDLC connection over a frame relay network, follow the procedures in Section 8.20.

Requirement

You can only set up CHDLC connections using one of the following Network Interface Cards:

- WANcontroller 614
- WANcontroller 618

- WANcontroller 622

8.19.2 Procedure

Follow these steps to create a new CHDLC connection:

1. Ensure that the MODEM CONNECT module exists by entering the following command:

   ```
   NCL> CREATE MODEM CONNECT
   ```

   Result: A message will tell you if the MODEM CONNECT module already exists; otherwise, it will be created.

2. Create a MODEM CONNECT LINE entity for the connection, specifying the hardware port on which the connection will exist, and the profile to be used for the line:

   ```
   NCL> CREATE MODEM CONNECT LINE line-name -
   _NCL> COMMUNICATION PORT port-name, PROFILE profile-name
   ```

   where:

   profile-name is either "NORMAL" or "DATEXP". The profile specifies the default value and permitted range for certain timers on the line. See the NCL online help for details of these timers.

   port-name is the name of the hardware port (see Section 8.3).

3. Set the MODEM CONTROL characteristic for the line:

   ```
   NCL> SET MODEM CONNECT LINE line-name MODEM CONTROL control
   ```

   where *control* is either FULL or NONE. FULL is the normal setting, and means that the line takes note of the modem control signals. Set it to NONE if the line is to ignore all modem control signals.

4. Enable the MODEM CONNECT LINE entity:

   ```
   NCL> ENABLE MODEM CONNECT LINE line-name
   ```

   Result: You have now established a synchronous line.

5. Ensure that the CHDLC module exists:

   ```
   NCL> CREATE CHDLC
   ```

6. Create a CHDLC data link for the circuit, and specify its type:

   ```
   NCL> CREATE CHDLC LINK link-name TYPE SYNCHRONOUS
   ```

   where *link-name* is a name identifying the data link. You should make this the same as the communication port name.

7. Associate the LINE created earlier with this data link:

   ```
   NCL> SET CHDLC LINK link-name -
   _NCL> LOWER LAYER ENTITY MODEM CONNECT LINE line-name
   ```

8. If your connection is a CHDLC connection to a DECbrouter™ 90, then before enabling the link you must set the SLARP TRANSMISSION TIMER to zero as follows:

   ```
   NCL> SET CHDLC LINK
   link-name SLARP TRANSMISSION TIMER 0
   ```

9. Enable the data link:

   ```
   NCL> ENABLE CHDLC LINK link-name
   ```

8.20 Creating a PPP or CHDLC Frame Relay Connection

8.20.1 Introduction

This section describes how to configure a PPP or CHDLC data link over a frame relay network. If you want to create a PPP link to a DEC WANrouter /Frame Relay 100/500, refer to Section 8.21.

Requirement

You can only set up PPP and CHDLC connections over frame relay on a WANcontroller 622 Network Interface card. This Card provides high-speed lines at T1/E1 (56 kbits/s to 2.048 Mbits/s).

8.20.2 How PPP and CHDLC Use Frame Relay

The PPP and CHDLC links use frame relay as the lower layer. When the link is created, it is associated with a frame relay channel and a frame relay connection:

- The channel identifies a single, physical line on the Network Interface Card.
- The connection provides access to the frame relay network over the channel. Up to 32 simultaneous connections can be set up on each channel.

To set up a logical path to the remote system, the DECNIS assigns the connection to a Permanent Virtual Circuit (PVC) in the frame relay network. The DECNIS will assign the connection to any available PVC, unless you define which PVC it should use. Defining a particular PVC is important if, for example, you need to set up a direct logical path to another vendor’s system.

8.20.3 Defining a PVC for the Frame Relay Connection

Each PVC within a frame relay network is identified by a Data Link Connection Identifier (DLCI). A DLCI is a 10-bit address that is assigned to the PVC by the frame relay network. All PVCs that are associated with one channel have unique DLCIs within that channel, although DLCIs are not necessarily unique throughout the frame relay network.

To define which PVC is used, you assign a PVC's DLCI to the connection. The connection always uses the PVC with the matching DLCI. Section 8.20.6 describes how to assign a DLCI to a connection.

8.20.4 Function of the Management Protocol

The DECNIS communicates with the frame relay network so that it knows which PVCs have been configured in the network. In addition, the frame relay network informs the DECNIS whenever a new PVC is configured or deleted.

This information is sent over the channel using one of the following management protocols:

- Joint/LMI
- ANSI T1.618 Annex D
- CCITT Q.933 Annex A

The protocol that the DECNIS uses must be the same as the protocol used by the frame relay device.

You can define the protocol when you create a frame relay channel (see Section 8.20.6).

8.20.5 Restrictions

The following restrictions apply to the frame relay implementation on the DECNIS:

- The BECN bit is ignored.
- The DE bit is always set ineligible.
- No flow control or committed information rate support is provided other than FECN to DECnet/OSI congestion bit mapping.
- Only DECNIS-to-DECNIS or DECNIS-to-WANrouter 500 operation is supported.
- Neither BOOTP nor MOP is supported across frame relay.
- There is no per DLC counter support.

- There is no CTF support.

8.20.6 Procedure

To create a PPP or CHDLC link over frame relay, follow these steps:

1. Ensure that the MODEM CONNECT module exists by entering the following command:

   ```
   NCL> CREATE MODEM CONNECT
   ```

   Result: A message will tell you if the MODEM CONNECT module already exists; otherwise, it will be created.

2. Create a MODEM CONNECT LINE entity for the connection, specifying the hardware port on which the connection will exist, and the profile to be used for the line.

   ```
   NCL> CREATE MODEM CONNECT LINE line-name -
   _NCL> COMMUNICATION PORT W622-slot-port, PROFILE profile-name
   ```

   where:

   profile-name is either "NORMAL" or "DATEXP". The profile specifies the default value and permitted range for certain timers on the line. See the NCL online help for details of these timers.

   W622-*slot-port* is the name of the hardware port (see Section 8.3).

3. Set the MODEM CONTROL characteristic for the line:

   ```
   NCL> SET MODEM CONNECT LINE line-name MODEM CONTROL control
   ```

   where *control* is either FULL or NONE. FULL is the normal setting, and means that the line takes note of the modem control signals. Set it to NONE if the line is to ignore all modem control signals.

4. Enable the MODEM CONNECT LINE entity:

   ```
   NCL> ENABLE MODEM CONNECT LINE line-name
   ```

   Result: You have now established a synchronous line.

5. Ensure that the FRBS module for frame relay exists:

   ```
   NCL> CREATE FRBS
   ```

6. Create an FRBS channel for the connection, and specify the management protocol used by the frame relay device:

   ```
   NCL> CREATE FRBS CHANNEL channel-name SPECIFICATION specification
   ```

   where *specification* is one of the following:

- JOINT
- ANSI-D
- CCITT
- NONE

Check that the protocol you specify is the same protocol used by the frame relay device. Use the NONE specification for back-to-back testing of two DECNIS systems without the need for a frame relay switch.

For simplicity, it is recommended that you make the *channel-name* the same as the *line-name*.

7. Now associate the LINE with this channel:

```
NCL> SET FRBS CHANNEL channel-name -
_NCL> PHYSICAL LINE MODEM CONNECT LINE line-name
```

8. Enable the channel:

```
NCL> ENABLE FRBS CHANNEL channel-name
```

9. Create a connection on the channel:

```
NCL> CREATE FRBS CHANNEL channel-name CONNECTION connection-name
```

   Note that the name of the connection should *not* be the same as the channel name. You can create up to 32 connections on each channel.

10. To ensure that the connection is not randomly assigned to a PVC, set the DLCI for the connection with the following command:

```
NCL> SET FRBS CHANNEL channel-name CONNECTION connection-name -
_NCL> PREFERRED DLCI dlci
```

   where *dlci* is the identifier for the connection. The PREFERRED DLCI characteristic can be cleared by setting the *dlci* to zero.

11. Enable the connection:

```
NCL> ENABLE FRBS CHANNEL channel-name CONNECTION connection-name
```

12. If you are setting up a CHDLC connection, ensure that the CHDLC module exists:

```
NCL> CREATE CHDLC
```

13. If you are setting up a PPP connection, ensure that the PPP module exists:

```
NCL> CREATE PPP
```

14. Create a data link for the circuit, and specify its type. For PPP data links:

    ```
    NCL> CREATE PPP LINK link-name TYPE SYNCHRONOUS
    ```

    where *link-name* is a name identifying the data link. You should make this the same as the communication port name.

    For CHDLC data links:

    ```
    NCL> CREATE CHDLC LINK link-name
    ```

    where *link-name* is a name identifying the data link. You should make this the same as the communication port name.

15. Associate the frame relay connection created earlier with this data link. For PPP data links:

    ```
    NCL> SET PPP LINK link-name LOWER LAYER ENTITY FRBS -
    _NCL> CHANNEL channel-name CONNECTION connection-name
    ```

    For CHDLC data links:

    ```
    NCL> SET CHDLC LINK link-name LOWER LAYER ENTITY FRBS -
    _NCL> CHANNEL channel-name CONNECTION connection-name
    ```

16. If your connection is a CHDLC connection to a DECbrouter 90, then before enabling the link you must set the SLARP TRANSMISSION TIMER to zero as follows:

    ```
    NCL> SET CHDLC LINK link-name SLARP TRANSMISSION TIMER 0
    ```

17. Enable the data link. For PPP data links:

    ```
    NCL> ENABLE PPP LINK link-name
    ```

    For CHDLC data links:

    ```
    NCL> ENABLE CHDLC LINK link-name
    ```

8.21 Creating a PPP Connection to a DEC WANrouter/Frame Relay 100/500

8.21.1 Restrictions

The following restrictions apply when you create a PPP link across a frame relay network to a DEC WANrouter/Frame Relay 100/500 (DEC WANrouter /Frame Relay):

- The DEC WANrouter/Frame Relay does not recognize PPP network control protocols used by the DECNIS. Section 8.21.2 describes how to set the Required Control Protocols attribute for the PPP link so that the DECNIS and DEC WANrouter/Frame Relay can interoperate.

Refer to Section 9.8.1 for more information about how network protocols are used on PPP routing circuits.

- Only IP and OSI network layer packets can be sent over a PPP link to a DEC WANrouter/Frame Relay.

8.21.2 Procedure

To create a connection to a DEC WANrouter/Frame Relay, complete the following:

1. Create a frame relay connection and PPP link by following the steps in Section 8.20.6 to step 14. Do not enable the data link.
2. Enter the following command:

   ```
   NCL> SET PPP LINK link-name REQUIRED CONTROL PROTOCOLS { }
   ```

   Note that there are no Required Control Protocols specified. This allows you to create a connection without network control protocol negotiation and IP and OSI network layer packets can be sent over the link.
3. Enable the data link:

   ```
   NCL> ENABLE PPP LINK link-name
   ```

8.22 Creating an SMDS Connection

This section describes how to configure an SMDS data link on a suitable synchronous line.

Requirement

You can only run SMDS connections on a WANcontroller 622 Network Interface Card.

8.22.1 Introduction to SMDS

SMDS (Switched Multimegabit Data Service) is a connectionless, public, packet-switched data service. SMDS provides features similar to those found in high-speed data networks such as LANs.

SMDS provides a datagram packet transfer. Each data unit contains the full destination SMDS address, so they are handled and switched separately without the prior establishment of a network connection.

8.22.1.1 SMDS Addressing

SMDS addresses consist of 64 bits, as shown in Figure 8–1. There are 60 bits of addressing information plus a 4-bit Address Type.

Figure 8–1 SMDS Address Field

Type	Address	Padding
4 bits	44 bits	16 bits

The Type subfield occupies the 4 most significant bits of the destination and source address fields of the SMDS Interface Protocol (SIP) Level 3 Protocol Data Unit (PDU). It contains the value 1100 (hexadecimal ‘C’) to indicate an individual address or the value 1110 (hexadecimal ‘E’) for a group address.

The DECNIS SMDS implementation uses the next 44 bits to represent the network address. These bits are represented by 11 binary coded decimal digits.

The least significant 16 bits of the address field are padded automatically with 1s by the DECNIS SMDS software to complete the 64-bit SMDS address.

For example, C1-12-34-56-78-90-FF-FF is a valid 64-bit SMDS address. You do not need to enter the padding values, so this address is specified by the string C1-12-34-56-78-90 in NCL commands.

8.22.2 Procedure

To create a new SMDS connection, follow these steps:

1. Ensure that the MODEM CONNECT entity exists by entering the following command:

   ```
   NCL> CREATE MODEM CONNECT
   ```

   Result: A message will tell you if the MODEM CONNECT entity already exists; otherwise, it will be created.

2. Create a MODEM CONNECT LINE entity for the connection, and specify the hardware port on which the connection will exist:

   ```
   NCL> CREATE MODEM CONNECT LINE line-name COMMUNICATION PORT W622-slot-port
   ```

where:

line-name	is the name of the line.
W622-*slot-port*	is the name of the hardware port (see Section 8.3).

3. Enable the MODEM CONNECT LINE entity:

   ```
   NCL> ENABLE MODEM CONNECT LINE line-name
   ```

   Result: You have now established a synchronous line.

4. Ensure that the SMDS entity exists by entering the following command:

   ```
   NCL> CREATE SMDS
   ```

   Result: A message will tell you if the SMDS entity already exists; otherwise, it will be created.

5. Create an SMDS STATION entity for the connection, and specify the hardware port on which the connection will exist.

   ```
   NCL> CREATE SMDS STATION station-name
   ```

   where *station-name* is a name to identify the SMDS station. Each SMDS station on the DECNIS must have a unique name.

6. Associate the line created earlier with the SMDS station:

   ```
   NCL> SET SMDS STATION station-name -
   _NCL> LOWER LAYER ENTITY MODEM CONNECT LINE line-name
   ```

7. Define the SMDS individual address for this Customer Premises Equipment (CPE):

   ```
   NCL> SET SMDS STATION station-name LOCAL ADDRESS local-address
   ```

   where *local-address* is the individual address assigned to this CPE at subscription time. The local address consists of a 10-digit decimal number following the prefix ‘C1’, with the pairs of digits separated by hyphens, for example, C1-12-34-56-78-90.

8. Define the SMDS group address which this CPE will communicate with. Messages will be multicast to all members of this group. You may specify an individual address if the communication will be with a single location.

   ```
   NCL> SET SMDS STATION station-name GROUP ADDRESS group-address
   ```

   where *group-address* is a 10-digit decimal number following the prefix ‘E1’, with the pairs of digits separated by hyphens, for example, E1-12-34-56-78-90.

9. Define the CRC option which you have subscribed to, using the command:

   ```
   NCL> SET SMDS STATION station-name CRC TYPE crc-type
   ```

   where *crc-type* is either 16-BIT or 32-BIT. The default value is 32-BIT.

10. Enable the SMDS STATION entity:

    ```
    NCL> ENABLE SMDS STATION station-name
    ```

8.22.3 Setting SMDS Heartbeat Poll Parameters

The heartbeat is a control message issued at regular intervals to confirm that the line connection is still operational. If the line does not transmit heartbeat poll messages, this indicates that the link is permanently available.

8.22.3.1 Turning Off Heartbeat Poll Messages

A HEARTBEAT TIMER value of 0 indicates that heartbeat poll messages will not be transmitted. This is the default.

If you do not wish to transmit heartbeat poll messages, leave the HEARTBEAT TIMER at its default value. Routing will then use the hello timer to determine availability.

8.22.3.2 Turning on Heartbeat Poll Messages

If you wish to issue heartbeat messages, set the HEARTBEAT TIMER to a nonzero value, and set the HEARTBEAT THRESHOLD characteristic. Refer to Section 8.22.3.4.

8.22.3.3 Responding to Heartbeat Poll Messages

The DECNIS will always respond to incoming heartbeat poll messages, even if you disable heartbeat polling by setting the timer to zero.

Note that if the heartbeat timer is set to zero, this indicates that the transmission of heartbeat poll messages is disabled. This is the default value, and the default heartbeat threshold is also 0.

8.22.3.4 Procedure

To set up heartbeat polling on your SMDS line, follow these steps:

1. Enter the heartbeat timer value:

   ```
   NCL> SET SMDS STATION station-name HEARTBEAT TIMER n, -
   ```

   where *n* is a decimal number between 0 and 255, and specifies the interval in seconds between heartbeat poll messages.

2. Enter the heartbeat threshold value:

   ```
   NCL> SET SMDS STATION station-name HEARTBEAT THRESHOLD m
   ```

   where *m* is a decimal number between 0 and 255, and specifies the number of heartbeat responses which may be missed before the link is considered to be down.

   Note that you must enter a nonzero value if you have entered a nonzero heartbeat timer value.

8.23 Data Compression on the DECNIS

The DECNIS supports data compression of routed packets. This section describes data compression on the DECNIS, and how to set it up.

8.23.1 Requirements

You can only set up data compression on the DEC WANcontroller 622 or DEC WANcontroller 622/HS (W622) Network Interface Cards, running the DEC HDLC data link protocol.

Both sides of an HDLC link need to be running data compression on the link in order for compression to take place.

8.23.2 Restrictions

8.23.2.1 Types of Packets Compressed

Only routed packets are compressed; bridged packets are not compressed. However, bridged packets are still bridged while compression is enabled; they simply are not compressed.

8.23.2.2 Interoperability

Because DECNIS data compression uses a proprietory algorithm, the DECNIS will only interoperate with another DECNIS over the links running data compression.

8.23.3 How Compression Affects Performance

The maximum line speeds supported when data compression is running on the line are as follows:

W622 Card	128 kbits/s
W622/HS (high speed) Card	384 kbits/s

Note that using higher speeds than these on lines running data compression may cause poor utilization of the available bandwidth, or even stop packets being transmitted.

8.23.4 Setting Up Data Compression

To set up data compression on an HDLC link, enter the following command:

```
NCL> SET HDLC LINK link-name PREFERRED COMPRESSION TYPES {DATAFLOW}
```

where *link-name* is the name of the HDLC LINK on which you want to run data compression.

Note that DATAFLOW is the only supported compression type.

8.23.5 Disabling Data Compression

To disable data compression, enter the following command:

```
NCL> SET HDLC LINK link-name PREFERRED COMPRESSION TYPES {}
```

Note that compression will not be disabled until the next time the HDLC link is set up—that is, the next time the link goes down, and comes up again.

8.23.6 Checking Compression Is Running

To check whether compression has been successfully negotiated, enter the following command when the `Protocol State` of the HDLC LOGICAL STATION is `Running`:

```
NCL> SHOW HDLC LINK link-name NEGOTIATED COMPRESSION TYPE
```

The value `None` indicates that compression is disabled. The value `Dataflow` indicates that compression is enabled.

8.23.7 Monitoring Data Compression

You can determine how much compression/expansion is taking place on an HDLC line by displaying four HDLC LINK LOGICAL CIRCUIT counters:

Counter	Purpose
Data Octets Sent	Counts the number of octets transmitted **before** compression.
Line Data Octets Sent	Counts the number of octets actually transmitted **after** compression.
Line Data Octets Received	Counts the number of octets actually received **before** decompression.
Data Octets Received	Counts the number of octets received **after** decompression.

8.23.7.1 Displaying the Compression Ratio

To display how much compression is taking place, divide DATA OCTETS by LINE DATA OCTETS. For example, to find the compression ratio for transmitted data, follow these steps:

1. Display the number of data octets sent before and after compression:

   ```
   NCL> SHOW HDLC LINK link-name LOGICAL STATION station-name -
   _NCL> DATA OCTETS SENT, LINE DATA OCTETS SENT
   ```

2. Divide DATA OCTETS SENT by LINE DATA OCTETS SENT. This gives the compression ratio.

Effect of Bridging on Compression Ratio

Note that because bridge traffic is not compressed, the compression ratio for a link carrying bridge traffic will be lower than if only compressed routed traffic were carried.

8.23.8 Using Data Compression on Noisy Links

Noisy and error-prone links normally have lower line utilization, as faulty packets need to be retransmitted. When such links are running data compression, line utilization is reduced still further. The reason for this is that if there is an error on a compressed link, the acknowledge timer must expire before recovery can take place. This is different from non-compressed DEC HDLC links, where a subsequent packet can initiate error recovery.

The acknowledge timer determines how long a link waits for a response to a message before retransmitting the message. The higher the value, the longer the delay before retransmitting the message.

To such minimize delays on compressed links, you should set the acknowledge timer to a value best suited to the line speed and packet size. It is likely that you will need to change the default of 3000ms, as it may well be too high for a compressed link.

8.23.8.1 Setting the Acknowledge Timer

Section 8.11.4 explains how to change the value of the HDLC acknowledge timer.

8.24 Deleting a Connection

8.24.1 Reason for Deleting

Delete a connection if you want to permanently prevent the DECNIS from communicating with the network through that connection.

8.24.2 Disabling Connections

Use the DISABLE command if you want to temporarily prevent the DECNIS from communicating with the network through a specific connection.

8.24.3 Before Deleting a Connection

Before you delete a connection, remove any routing, bridging, or X.25 functions that use the connection. For example, you should delete any routing circuits using the connection.

If you cannot delete a connection, but only disable it, as in the case of CSMA/CD and SMDS connections, then first disable the functions using the connection.

8.24.4 Disabling a CSMA/CD Connection

You cannot delete a CSMA-CD station entity on the DECNIS.

Disable a CSMA/CD connection by issuing the following command (see Section 8.25 if you do not know the name of the CSMA-CD STATION):

```
NCL> DISABLE CSMA-CD STATION station-name
```

8.24.5 Deleting a VCP Connection

Follow these steps to delete a VCP connection:

1. Disable the CSMA-CD STATION entity for the connection.

   ```
   NCL> DISABLE CSMA-CD STATION station-name
   ```

2. Disable the MODEM CONNECT LINE entity for the connection.

   ```
   NCL> DISABLE MODEM CONNECT LINE line-name
   ```

3. Delete the MODEM CONNECT LINE entity for the connection.

   ```
   NCL> DELETE MODEM CONNECT LINE line-name
   ```

8.24.6 Deleting an HDLC Connection

Follow these steps to delete an HDLC connection:

1. Disable the logical station associated with the circuit:

   ```
   NCL> DISABLE HDLC LINK link-name LOGICAL STATION station-name
   ```

2. Delete this logical station:

   ```
   NCL> DELETE HDLC LINK link-name LOGICAL STATION station-name
   ```

3. Disable the HDLC link associated with this circuit:

   ```
   NCL> DISABLE HDLC LINK link-name
   ```

4. Delete this link:

   ```
   NCL> DELETE HDLC LINK link-name
   ```

5. Disable the MODEM CONNECT LINE entity for the connection. See Section 8.25 if you do not know the name of the line.

   ```
   NCL> DISABLE MODEM CONNECT LINE line-name
   ```

6. Delete the MODEM CONNECT LINE entity for the connection.

   ```
   NCL> DELETE MODEM CONNECT LINE line-name
   ```

8.24.7 Deleting a DDCMP Connection

Follow these steps to delete a DDCMP connection:

1. Disable the logical station associated with the circuit:

   ```
   NCL> DISABLE DDCMP LINK link-name LOGICAL STATION station-name
   ```

2. Delete this logical station:

   ```
   NCL> DELETE DDCMP LINK link-name LOGICAL STATION station-name
   ```

3. Disable the DDCMP link associated with this circuit:

   ```
   NCL> DISABLE DDCMP LINK link-name
   ```

4. Delete this link:

   ```
   NCL> DELETE DDCMP LINK link-name
   ```

5. Disable the MODEM CONNECT LINE entity for the connection. See Section 8.25 if you do not know the name of the line.

   ```
   NCL> DISABLE MODEM CONNECT LINE line-name
   ```

6. Delete the MODEM CONNECT LINE entity for the connection.

   ```
   NCL> DELETE MODEM CONNECT LINE line-name
   ```

8.24.8 Deleting an ATM Connection

To delete an ATM connection, disable the module entities associated with that ATM connection and then delete those entities.

Follow these steps to delete an ATM connection:

1. Disable the Multiplexed Interface Logical Channel:

   ```
   NCL> DISABLE MULTIPLEXED INTERFACE LINE line-name -
   _NCL> LOGICAL CHANNEL channel-name
   ```

2. Disable the Multiplexed Interface Line:

   ```
   NCL> DISABLE MULTIPLEXED INTERFACE LINE line-name
   ```

3. Disable the ATM Connection Management Line:

   ```
   NCL> DISABLE ATM CONNECTION MANAGEMENT LINE line-name
   ```

4. Disable the ATM Multiprotocol Encapsulation Link:

   ```
   NCL> DISABLE ATM MULTIPROTOCOL ENCAPSULATION LINK link-name
   ```

5. Delete the Multiplexed Interface Logical Channel:

   ```
   NCL> DELETE MULTIPLEXED INTERFACE LINE line-name -
   _NCL> LOGICAL CHANNEL channel-name
   ```

6. Delete the Multiplexed Interface Line:

   ```
   NCL> DELETE MULTIPLEXED INTERFACE LINE line-name
   ```

7. Delete the ATM Connection Management PVC:

   ```
   NCL> DELETE ATM CONNECTION MANAGEMENT LINE line-name PVC pvc-name
   ```

8. Delete the ATM Connection Management Line:

   ```
   NCL> DELETE ATM CONNECTION MANAGEMENT LINE line-name
   ```

9. Delete the ATM Multiprotocol Encapsulation Link:

   ```
   NCL> DELETE ATM MULTIPROTOCOL ENCAPSULATION LINK link-name
   ```

8.24.9 Deleting a PPP Connection

Follow these steps to delete a PPP connection:

1. Disable the PPP link associated with the circuit:

   ```
   NCL> DISABLE PPP LINK link-name
   ```

2. Delete this link:

   ```
   NCL> DELETE PPP LINK link-name
   ```

3. Disable the MODEM CONNECT LINE entity for the connection. See Section 8.25 if you do not know the name of the line.

   ```
   NCL> DISABLE MODEM CONNECT LINE line-name
   ```

4. Delete the MODEM CONNECT LINE entity for the connection.

   ```
   NCL> DELETE MODEM CONNECT LINE line-name
   ```

8.24.10 Deleting a PPP Connection Over Frame Relay

Follow these steps to delete a PPP connection over a frame relay network:

1. Disable the PPP link associated with the circuit:

   ```
   NCL> DISABLE PPP LINK link-name
   ```

2. Delete the link:

   ```
   NCL> DELETE PPP LINK link-name
   ```

3. Disable the frame relay connection:

   ```
   NCL> DISABLE FRBS CHANNEL channel-name CONNECTION connection-name
   ```

4. Delete the frame relay connection:

   ```
   NCL> DELETE FRBS CHANNEL channel-name CONNECTION connection-name
   ```

5. Disable and delete the channel associated with the connection:

   ```
   NCL> DISABLE FRBS CHANNEL channel-name
   NCL> DELETE FRBS CHANNEL channel-name
   ```

6. Disable the MODEM CONNECT LINE entity for the connection. See Section 8.25 if you do not know the name of the line.

   ```
   NCL> DISABLE MODEM CONNECT LINE line-name
   ```

7. Delete the MODEM CONNECT LINE entity for the connection.

   ```
   NCL> DELETE MODEM CONNECT LINE line-name
   ```

8.24.11 Disabling an SMDS Connection

You cannot delete an SMDS station entity on the DECNIS; you can only disable it.

Follow these steps to disable an SMDS connection:

1. Disable the SMDS STATION entity for the connection. See Section 8.25.17 if you do not know the name of the SMDS STATION.

   ```
   NCL> DISABLE SMDS STATION station-name
   ```

2. Disable the MODEM CONNECT LINE entity for the connection.

   ```
   NCL> DISABLE MODEM CONNECT LINE line-name
   ```

8.25 Identifying Management Entities from the Hardware Port

8.25.1 Introduction

This section describes how to find the names of the following entities, given that you know the hardware port name:

- CSMA-CD STATION
- MODEM CONNECT LINE
- MULTIPLEXED INTERFACE LINE
- HDLC LINK
- HDLC LINK LOGICAL STATION
- DDCMP LINK
- DDCMP LOGICAL STATION
- FRBS CHANNEL
- FRBS CHANNEL CONNECTION
- ATM CONNECTION MANAGEMENT LINE
- ATM CONNECTION MANAGEMENT LINE PVC
- ATM MULTIPROTOCOL ENCAPSULATION LINK
- PPP LINK
- CHDLC LINK
- SMDS STATION

8.25.2 CSMA-CD STATION Name

To display the name of the CSMA-CD STATION entity, issue the following command:

```
NCL> SHOW CSMA-CD STATION * ALL IDENTIFIERS, WITH COMMUNICATION PORT -
_NCL> port-name
```

where *port-name* is the name of the associated hardware port, for example L601-4-0 (see Section 8.3).

8.25.3 FDDI STATION Name

To display the name of the FDDI STATION entity, issue the following command:

```
NCL> SHOW FDDI STATION * ALL IDENTIFIERS, WITH COMMUNICATION PORT -
_NCL> port-name
```

where *port-name* is the name of the associated hardware port, for example F621-6-1 (see Section 8.3).

8.25.4 MODEM CONNECT LINE Name

To display the name of the MODEM CONNECT LINE entity, issue the following command:

```
NCL> SHOW MODEM CONNECT LINE * ALL IDENTIFIERS, WITH -
_NCL> COMMUNICATION PORT port-name
```

where *port-name* is the name of the associated hardware port, for example W622-4-0 (see Section 8.3).

8.25.5 MULTIPLEXED INTERFACE LINE Name

To display the name of the MULTIPLEXED INTERFACE LINE entity, issue the following command:

```
NCL> SHOW MULTIPLEXED INTERFACE LINE * ALL IDENTIFIERS, WITH -
_NCL> COMMUNICATIONS PORT port-name
```

where *port-name* is the name of the associated hardware port, for example W631-8-0 (see Section 8.3).

8.25.6 HDLC LINK Name

To display the name of the HDLC LINK entity, issue the following command:

```
NCL> SHOW HDLC LINK * ALL IDENTIFIERS, WITH PHYSICAL LINE -
_NCL> MODEM CONNECT LINE line-name
```

where *line-name* is the name of the associated line, as discovered in Section 8.25.4.

8.25.7 HDLC LINK LOGICAL STATION Name

To display the name of the HDLC LINK LOGICAL STATION entity, issue the following command:

```
NCL> SHOW HDLC LINK link-name LOGICAL STATION * -
_NCL> ALL IDENTIFIERS
```

where *link-name* is the name of the associated HDLC LINK, as discovered in Section 8.25.6.

8.25.8 DDCMP LINK Name

To display the name of the DDCMP LINK entity, issue the following command:

```
NCL> SHOW DDCMP LINK * ALL IDENTIFIERS, WITH PHYSICAL LINE -
_NCL> MODEM CONNECT LINE line-name
```

where *line-name* is the name of the associated line, as discovered in Section 8.25.4.

8.25.9 DDCMP LOGICAL STATION Name

To display the name of the DDCMP LINK LOGICAL STATION entity, issue the following command:

```
NCL> SHOW DDCMP LINK link-name LOGICAL STATION * -
_NCL> ALL IDENTIFIERS
```

where *link-name* is the name of the associated HDLC LINK, as discovered in Section 8.25.6.

8.25.10 FRBS CHANNEL Name

To display the name of the FRBS CHANNEL entity, issue the following command:

```
NCL> SHOW FRBS CHANNEL * ALL IDENTIFIERS, WITH PHYSICAL LINE -
_NCL> MODEM CONNECT LINE line-name
```

where *line-name* is the name of the associated line, as discovered in Section 8.25.4.

8.25.11 FRBS CHANNEL CONNECTION Name

To display the name of the FRBS CHANNEL CONNECTION entity, issue the following command:

```
NCL> SHOW FRBS CHANNEL channel-name CONNECTION * -
_NCL> ALL IDENTIFIERS
```

where *channel-name* is the name of the associated channel, as discovered in Section 8.25.10.

8.25.12 ATM Connection Management Line

To display the name of the ATM CONNECTION MANAGEMENT LINE entity, issue the following command:

```
NCL> SHOW ATM CONNECTION MANAGEMENT LINE * ALL IDENTIFIERS, WITH -
_NCL LOWER LAYER ENTITY -
_NCL> MULTIPLEXED INTERFACE LINE line-name
```

where *line-name* is name of the associated Multiplexed Interface line, as discovered in Section 8.25.5.

8.25.13 ATM Connection Management Line PVC

To display the name of the ATM CONNECTION MANAGEMENT LINE PVC entity, issue the following command:

```
NCL> SHOW ATM CONNECTION MANAGEMENT LINE line-name -
_NCL> PVC * ALL IDENTIFIERS
```

where *line-name* is name of the associated ATM Connection Management line, as discovered in Section 8.25.12.

8.25.14 ATM Multiprotocol Encapsulation Link

To display the name of the ATM MULTIPROTOCOL ENCAPSULATION LINK entity, issue the following command:

```
NCL> SHOW ATM MULTIPROTOCOL ENCAPSULATION LINK * ALL IDENTIFIERS, WITH -
_NCL LOWER LAYER ENTITY -
_NCL> ATM  CONNECTION MANAGEMENT line-name PVC pvc-name
```

where *line-name* is the ATM Connection Management line, as discovered in Section 8.25.12 and *pvc-name* is the PVC, as discovered in Section 8.25.13.

8.25.15 PPP LINK Name

The procedure that you use for determining the name of the PPP LINK entity depends on whether you are using frame relay:

- If you are **not** using frame relay, issue the following command to display the name of the PPP LINK entity:

  ```
  NCL> SHOW PPP LINK * ALL IDENTIFIERS, WITH LOWER LAYER ENTITY -
  _NCL> MODEM CONNECT LINE line-name
  ```

  where *line-name* is the name of the associated line, as discovered in Section 8.25.4.

- If you are using frame relay, issue the following command to display the name of the PPP LINK entity:

  ```
  NCL> SHOW PPP LINK * ALL IDENTIFIERS, WITH LOWER LAYER ENTITY -
  _NCL> FRBS CHANNEL channel-name CONNECTION connection-name
  ```

  where *channel-name* is the name of the associated channel, as discovered in Section 8.25.10, and *connection-name* is the name of the connection, as discovered in Section 8.25.11.

8.25.16 CHDLC LINK Name

The procedure that you use for finding the name of the CHDLC LINK entity is nearly the same as that for finding the name of the PPP LINK entity; the only difference is that the SHOW commands start:

```
NCL> SHOW CHDLC LINK
```

rather than SHOW PPP LINK.

8.25.17 SMDS STATION Name

To display the name of the SMDS STATION entity, issue the following command:

```
NCL> SHOW SMDS STATION * NAME, WITH -
_NCL> LOWER LAYER ENTITY MODEM CONNECT LINE line-name
```

where *line-name* is the name of the associated line.

8.26 Creating MOP Circuits for Loopback Testing

8.26.1 Introduction

This section describes how to set up a MOP (Maintenance Operations Protocol) circuit on a data link, and configure it so that it can be used to initiate loopback tests.

The DECNIS online Problem Solving manual explains how to perform loopback testing to check the communication path between the DECNIS and a remote node. However, before you can use the loopback commands described in that manual, you must create a MOP circuit for the link, and include LOOP REQUESTER as one of its functions.

This is done automatically when you use the DECNIS configurator (text-based or Windows) to create new data links.

ATM Permanent and CHDLC Data Links

You cannot set up a MOP circuit on an ATM Permanent or CHDLC data link.

8.26.2 Procedure

For data links that you create using interactive NCL commands, you must follow these steps to allow loopback testing on those data links.

1. Create a MOP circuit, and specify the type of data link over which it will run:

   ```
   NCL> CREATE NODE decnis MOP CIRCUIT circuit-name TYPE type
   ```

   where *decnis* is the name of the DECNIS, and *type* is one of:

   - CSMA-CD
   - DDCMP™
   - FDDI
   - HDLC (also used for PPP MOP circuits)
   - LAPB

   circuit-name identifies the MOP circuit. For simplicity, this name should be the same as the name of any routing circuit that uses the data link.

2. Specify the data link that the circuit will use:

   - For CSMA/CD data links:

     ```
     NCL> SET NODE decnis MOP CIRCUIT circuit-name -
     _NCL> LINK NAME CSMA-CD STATION station-name
     ```

   - For DDCMP data links:

     ```
     NCL> SET NODE decnis MOP CIRCUIT circuit-name -
     _NCL> LINK NAME DDCMP LINK link-name -
     _NCL> LOGICAL STATION station-name
     ```

   - For FDDI data links:

     ```
     NCL> SET NODE decnis MOP CIRCUIT circuit-name -
     _NCL> LINK NAME FDDI STATION station-name
     ```

   - For HDLC data links:

     ```
     NCL> SET NODE decnis MOP CIRCUIT circuit-name -
     _NCL> LINK NAME HDLC LINK link-name -
     _NCL> LOGICAL STATION station-name
     ```

   - For LAPB data links:

     ```
     NCL> SET NODE decnis MOP CIRCUIT circuit-name -
     _NCL> LINK NAME LAPB LINK link-name
     ```

3. For PPP data links:

```
NCL> SET NODE decnis MOP CIRCUIT circuit-name -
_NCL> LINK NAME PPP LINK link-name
```

   Note that the PPP MOP circuit is of type HDLC.

4. Enable the LOOP REQUESTER function on the MOP circuit:

```
NCL> ENABLE NODE decnis MOP CIRCUIT circuit-name FUNCTION {LOOP REQUESTER}
```

8.27 Using SNMP to Manage Connections

The following sections describe some of the MIB variables related to interfaces, and their meaning in DECNIS terms.

ifDescr

MIB: MIB-II, *MIB group:* Interfaces, *MIB table:* ifTable

For each of the DECNIS interfaces, the value of the ifDesc MIB variable is the name of the data link on that interface. If you used the DECNIS configurator to set up the data links, the name will be the same as the hardware port (see Section 8.3).

ifPhysAddress

MIB: MIB-II, *MIB group:* Interfaces, *MIB table:* ifTable

For synchronous interfaces (apart from VCP), this variable will appear as a null string.

ifType

MIB: MIB-II, *MIB group:* Interfaces, *MIB table:* ifTable

This value will appear as follows:

DECNIS Interface Type	ifType Value
ATM MULTIPROTOCOL ENCAPSULATION	AAL5
CHDLC	ppp
CSMA/CD	ethernet-csmacd
DDCMP	other
FDDI	fddi
HDLC	propPointToPointSerial
LAPB	lapb

DECNIS Interface Type	ifType Value
PPP	ppp
SMDS	sip
VCP	ethernet-csmacd

ifAdminStatus: Enabling and Disabling Connections

MIB: MIB-II, *MIB group:* Interfaces, *MIB table:* ifTable

Set this to "up" to enable a DECNIS data link; set it to "down" to disable it. You cannot set it to "testing".

When the interface is an FDDI data link, setting it to "up" has the same effect as setting the MACAction variable in the FDDI MIB to "enableLLCService" (see below).

FDDI Connections: SMTTable

MIB: FDDI MIB, *MIB group:* SMT, *MIB table:* SMTTable

The DECNIS implements all the entries in this table as read-only.

FDDI Connections: MACTable

MIB: FDDI MIB, *MIB group:* MAC, *MIB table:* MACTable

The DECNIS implements the following entries in this table as read-only:

MACTMaxGreatestLowerBound
MACPathsRequested
MACCurrentFrameStatus

FDDI Connections: MACTReq

MIB: FDDI MIB, *MIB group:* MAC, *MIB table:* MACTable

You can only set this variable when the data link is disabled (by setting ifAdminStatus or MACAction to the appropriate value, or by using NCL to disable the FDDI STATION LINK entity).

MACAction: Enabling and Disabling FDDI Connections

MIB: FDDI MIB, *MIB group:* MAC, *MIB table:* MACTable

Set this to "enableLLCService" to enable the data link (equivalent to enabling the FDDI STATION LINK entity). Set it to "disableLLCService" to disable the data link. These are the only two values supported in the DECNIS.

FDDI Connections: PORTTable

MIB: FDDI MIB, *MIB group:* PORT, *MIB table:* PORTTable

The DECNIS implements the following entries in this table as read-only:

PORTConnectionPolicies
PORTPathsRequested
PORTLerAlarm

You can set the following variables on the DECNIS only when the port is disabled (by setting PORTAction to the appropriate value, or by using NCL to disable the FDDI STATION PHY PORT entity):

PORTMACLoopTime
PORTTBMax
PORTLerCutoff

PORTAction: Disabling an FDDI Port

MIB: FDDI MIB, *MIB group:* PORT, *MIB table:* PORTTable

Set this to "enablePORT" to enable an FDDI port; set it to "disablePORT" to disable it.

These are the only two values that you can set on the DECNIS.

ATTACHMENTInsertPolicy

MIB: FDDI MIB, *MIB group:* ATTACHMENT, *MIB table:* ATTACHMENTTable

The DECNIS implements this variable as read-only.

eFddiMACTable

MIB: DEC Vendor MIB elanext V2.7, *MIB group:* eFddi, *MIB table:* eFddiMACTable

You can only set the following variables when the data link is disabled (by setting ifAdminStatus or MACAction to the appropriate value, or by using NCL to disable the FDDI STATION LINK entity):

eMACRingPurgerEnable
eMACRestrictedTokenTimeout

eFDXEnable

MIB: DEC Vendor MIB elanext V2.7, *MIB group:* eFddi, *MIB table:* eFddiMACFDXTable

This variable is not supported in the DECNIS.

Part II

Managing Routing

This part contains information on managing routing on the DECNIS.

It contains the following chapters:

- Chapter 9 describes how to set up routing circuits, and discusses routing security.
- Chapter 10 describes how to use the DECNIS for IP routing.
- Chapter 11 describes how the DECNIS implements IP route propagation and filtering, for all supported IP protocols except BGP.
- Chapter 12 describes how the DECNIS implements BGP route propagation and filtering.
- Chapter 13 describes how to set up and use IP packet filtering on the DECNIS.
- Chapter 14 describes how to set up the DECNIS as an IP multicast router.
- Chapter 15 describes how to set up the IP Standby facility on the DECNIS.
- Chapter 16 describes how to use the DECNIS for DECnet/OSI routing.
- Chapter 17 describes how to use the DECNIS for routing AppleTalk and NetWare IPX protocols.

9

Managing Routing: General

9.1 Introduction

This chapter describes how to manage those aspects of routing that are independent of the protocols being routed.

This chapter discusses the following aspects of routing:

- Setting up CSMA/CD, FDDI, HDLC, DDCMP, PPP, CHDLC, SMDS, ATM Permanent and X.25 routing circuits
- Routing level
- Routing security

9.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

Note

For any tasks that require you to disable a routing circuit, ensure that you do not disable the routing circuit that you are using to manage the DECNIS, unless there is an alternative route between the host on which you issue the NCL commands and the DECNIS.

9.1.2 Entities Associated with a Routing Circuit

Figure 9–1, Figure 9–2, Figure 9–3 and Figure 9–4 show the entities required by each type of routing circuit.

9.2 Creating ROUTING

In order to create routing entities, the ROUTING module must exist. To ensure that the ROUTING module exists, enter the following command:

```
NCL> CREATE ROUTING
```

Result: A message will tell you if the ROUTING module entity already exists; otherwise, it will be created.

9.3 Adding a CSMA/CD Routing Circuit

9.3.1 Introduction

This section describes how to create a CSMA/CD routing circuit. The information in this section applies also to creating routing circuits on VCP data links (which use a CSMA/CD station on a synchronous line).

Before you begin, you must create a CSMA/CD STATION entity, as described in Section 8.7.2 and Section 8.9.

9.3.2 Procedure

Create a new CSMA/CD routing circuit as follows:

1. Create a routing circuit of type CSMA-CD:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name TYPE CSMA-CD
   ```

2. Specify the CSMA-CD STATION entity that the circuit will use as its data link:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> DATA LINK ENTITY CSMA-CD STATION station-name
   ```

3. Enable the circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

9.4 Adding an FDDI Routing Circuit

9.4.1 Introduction

This section describes how to create an FDDI routing circuit. Before you begin, you must create an FDDI STATION entity, as described in Section 8.8.2.

9.4.2 Procedure

Create a new FDDI routing circuit as follows:

1. Create a routing circuit, specifying that it is of type FDDI:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name TYPE FDDI
   ```

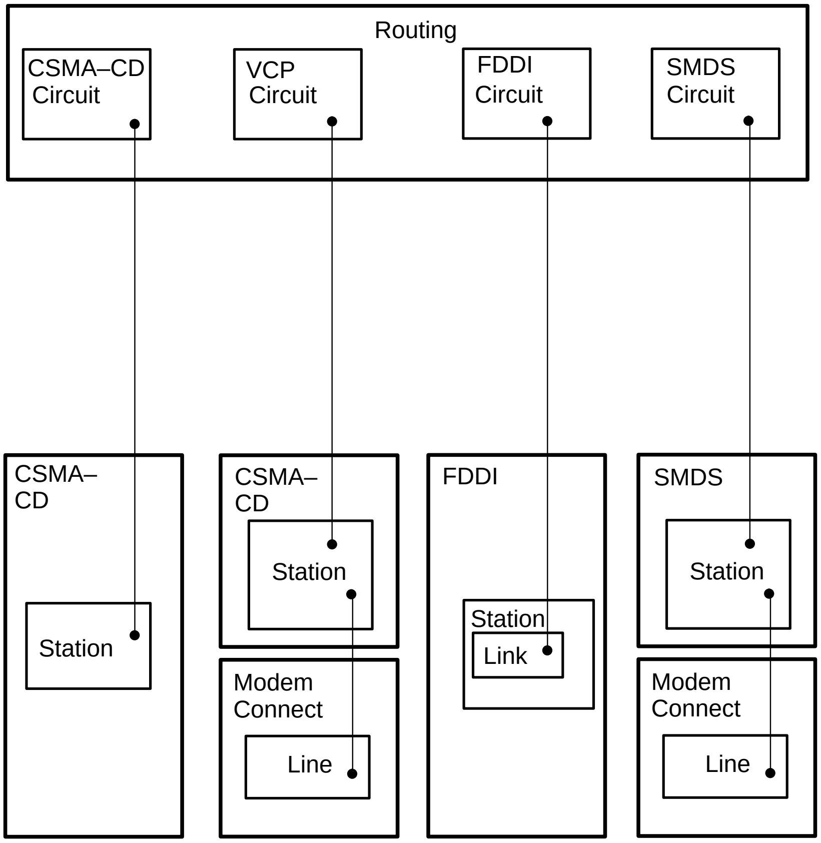
2. Specify the FDDI STATION entity that the circuit will use as its data link:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> DATA LINK ENTITY FDDI STATION station-name
   ```

3. Enable the circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

Figure 9–1 Entities Required by LAN, VCP and SMDS Routing Circuits

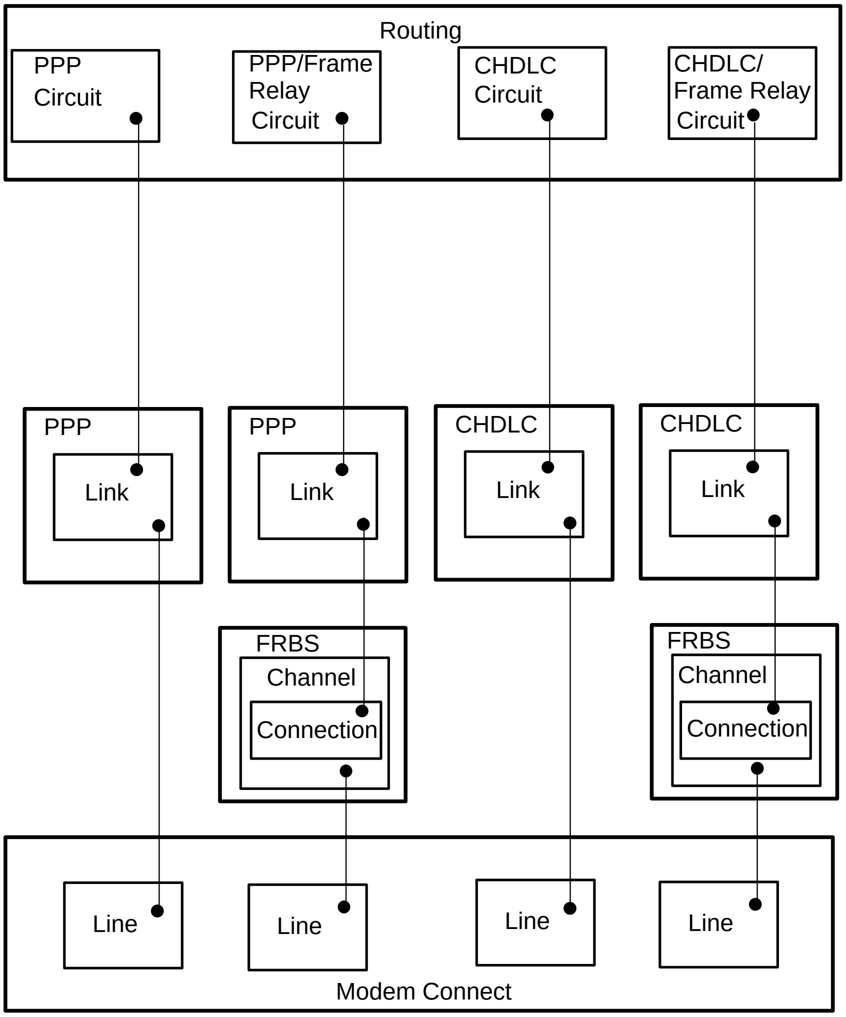


An association between two entities (indicated by a line connecting them) is made by specifying the name of the lower entity as a characteristic value of the upper entity.

For example: a routing circuit of type FDDI specifies as its DATA LINK entity the name of the FDDI link.

CBN–0084–94–I

Figure 9–2 Entities Required by PPP and Routing Circuits

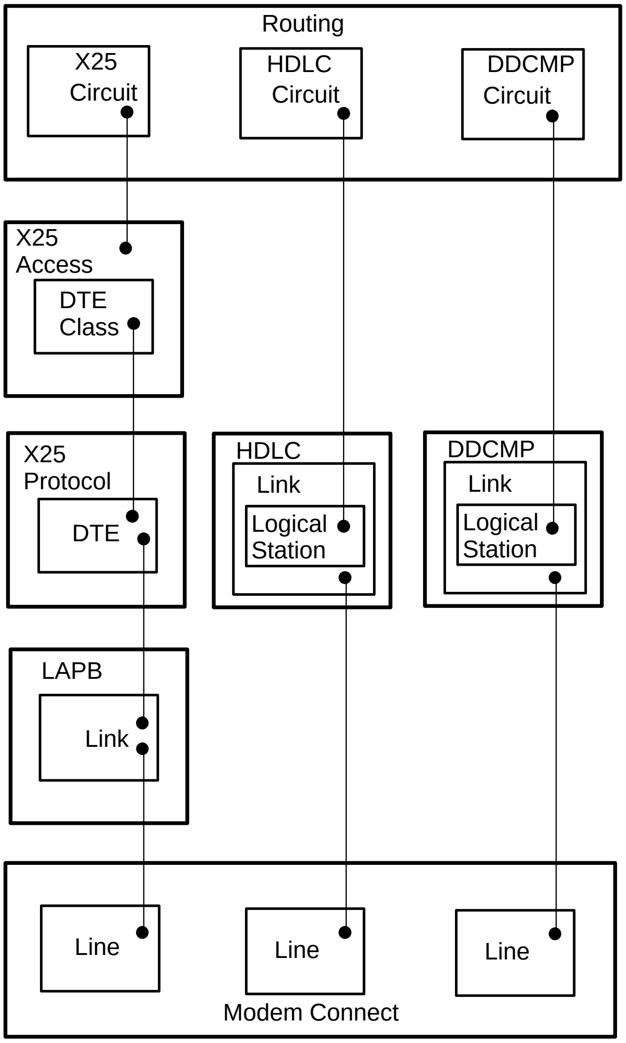


An association between two entities (indicated by a line connecting them) is made by specifying the name of the lower entity as a characteristic value of the upper entity.

For example: a routing circuit of type PPP specifies as its DATA LINK entity the name of the PPP Link.

CBN–0085–94–I

Figure 9–3 Entities Required by X.25, HDLC, and DDCMP Routing Circuits

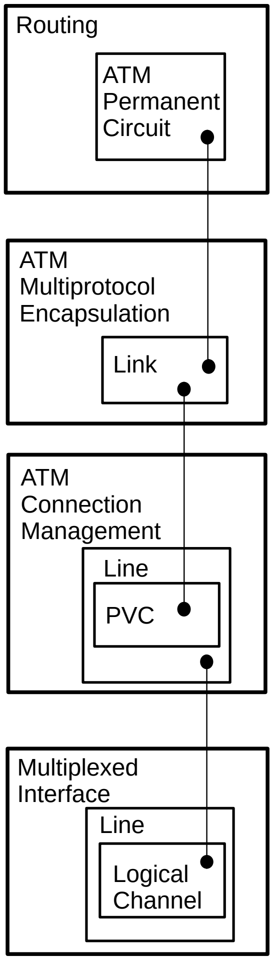


An association between two entities (indicated by a line connecting them) is made by specifying the name of the lower entity as a characteristic value of the upper entity.

For example: a routing circuit of type HDLC specifies as its DATA LINK entity the name of the HDLC Logical Station.

CBN–0086–94–I

Figure 9–4 Entities Required by ATM Permanent Routing Circuits



An association between two entities (indicated by a line connecting them) is made by specifying the name of the lower entity as a characteristic value of the upper entity.

For example, a routing circuit that uses ATM point–to–point service specifies as its DATA LINK entity the name of the ATM Multiprotocol Encapsulation Link.

CBN–0001–95–I

9.5 Adding an HDLC or DDCMP Routing Circuit

9.5.1 Introduction

This section describes how to create an HDLC or DDCMP routing circuit. Before you begin, you must create an HDLC or DDCMP data link, as described in Section 8.10 and Section 8.15.

9.5.2 Procedure

Create a new HDLC or DDCMP routing circuit as follows:

1. Create a routing circuit, specifying its type:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name TYPE type
   ```

   where *type* is either HDLC or DDCMP.

2. Specify the link and logical station that this circuit is to use as its data link:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> DATA LINK ENTITY type LINK link-name -
   _NCL> LOGICAL STATION station-name
   ```

3. Enable the circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

9.6 ATM Permanent Routing Circuits

9.6.1 Introduction

ATM Point-to-Point routing circuits are protocol specific. This means that you must define the network protocols that will be routed over a circuit. You can do this using the NCL commands described in Section 9.7.

Note that there is no support for ATM circuits in the DECNIS text-based configurator. However, ATM circuits are supported in the clearVISN DECNIS configurator.

9.6.2 Defining Protocols for ATM Permanent Circuits

The protocols that you define depend on the configuration of your network and the type of data routed over the circuit. You can specify one or more of the following:

- **IP** for IP data and for IP routing control packets. For example, EGP and RIP.

- **ISO8473** for DECnet/OSI data and for DECnet/OSI routing control packets. For example, integrated IS-IS (ISO 10589/RFC 1195) and ES-IS (ISO 9542).
- **DECNET PHASE IV** for DECnet Phase IV data and DECnet Phase IV routing control packets.
- **NETWARE IPX** for NetWare IPX packets.

For example, if the remote system uses IP routing, you must define IP as the network protocol to be routed over the circuit between the DECNIS and the remote system.

9.6.3 Default Protocols

If you do not define a protocol for the circuit, the system sets up IP and ISO8473 on the circuit by default.

9.6.4 ATM Permanent Point-to-Point Circuits and DECnet Phase IV

If the DECNIS and remote system use DECnet Phase IV routing, refer to Appendix F.

The tables in that appendix list possible combinations of routing algorithms used by the DECNIS and the remote system and show whether you need to provide manual routing information for the circuit. The tables also indicate which network protocols you should define for each combination.

Note that for DECnet Phase IV routing between two DECnet/OSI systems, you need to specify both DECNET PHASE IV and ISO8473 as network protocols. This is because DECnet Phase IV data is translated to DECnet/OSI data before it is routed over the circuit between the two systems.

9.7 Adding an ATM Permanent Routing Circuit

9.7.1 Introduction

This section describes how to set up an ATM Permanent routing circuit. Before you begin, you must create an ATM connection by following the procedures in Section 8.6.

9.7.2 Procedure

Create a new ATM Permanent routing circuit as follows:

1. Create a routing circuit, specifying that it is of type ATM Permanent:

```
NCL> CREATE ROUTING CIRCUIT circuit-name TYPE ATM PERMANENT
```

2. Associate the routing circuit with the ATM Multiprotocol Encapsulation Link:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> DATA LINK ENTITY ATM MULTIPROTOCOL ENCAPSULATION LINK link-name
   ```

3. Specify the network protocols to run over the circuit. If you do not specify a protocol, IP and ISO8473 are set up on the circuit automatically. Note that the remote system must support the protocols that you select.

   Enter the following command:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> NETWORK PROTOCOLS {protocol1, protocol2, protocol3}
   ```

   where *protocol1*, *protocol2*, and *protocol3* are the protocols that you require. You can specify one or more of the following:

   - IP
   - ISO8473
   - DECNET PHASE IV
   - NETWARE IPX

   For example:

   ```
   NCL> SET ROUTING CIRCUIT W631-7-0 NETWORK PROTOCOL {IP,DECNET PHASE IV}
   ```

4. Enable the circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

9.8 PPP Routing Circuits

9.8.1 Defining Protocols for PPP Circuits

PPP routing circuits are protocol specific. This means that you must define the network protocols that will be routed over a circuit. This is described in Section 9.9.

The protocols that you define depend on the configuration of your network and the type of data routed over the circuit. You can specify one or more of the protocols listed in Section 9.6.2.

For example, if the remote system uses IP routing, you must define IP as the network protocol to be routed over the circuit between the DECNIS and the remote system.

9.8.2 Default Protocols

If you do not define a protocol for the circuit, the system sets up IP and ISO8473 on the circuit by default.

9.8.3 PPP Circuits and DECnet Phase IV

If the DECNIS and remote system use DECnet Phase IV routing, refer to Appendix F.

The tables in that appendix list possible combinations of routing algorithms used by the DECNIS and the remote system and show whether you need to provide manual routing information for the circuit. The tables also indicate which network protocols you should define for each combination.

Note that for DECnet Phase IV routing between two DECnet/OSI systems, you need to specify both DECNET PHASE IV and ISO8473 as network protocols. This is because DECnet Phase IV data is translated to DECnet/OSI data before it is routed over the circuit between the two systems.

9.8.4 System Resources

Only required protocols should be set up on each circuit. If you define a circuit to route unused protocols, it could waste system resources.

For example, when a circuit is set up for the network protocol ISO8473, the DECNIS and remote system continuously acknowledge each other by sending ‘hello’ packets using this protocol. Even if the remote system does not use ISO8473, this exchange still takes place. This means that unnecessary packets are being sent across the circuit.

9.9 Adding a PPP Routing Circuit

9.9.1 Introduction

This section describes how to create a PPP routing circuit. Before you begin, you must create a PPP data link by following the procedures in Section 8.18.

9.9.2 Procedure

Create a new PPP routing circuit as follows:

1. Create a routing circuit, specifying that it is of type PPP:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name TYPE PPP
   ```

2. Specify the link that this circuit is to use as its data link:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> DATA LINK ENTITY PPP LINK link-name
   ```

3. Specify the network protocols to run over the circuit. If you do not specify a protocol, IP and ISO8473 are set up on the circuit automatically. Note that the remote system must support the protocols that you select.

   Enter the following command:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> NETWORK PROTOCOLS {protocol1, protocol2, protocol3}
   ```

   where *protocol1*, *protocol2*, and *protocol3* are the protocols that you require. You can specify one or more of the following:

   - IP
   - ISO8473
   - DECNET PHASE IV
   - NETWARE IPX

   For example:

   ```
   NCL> SET ROUTING CIRCUIT W622-4-1 NETWORK PROTOCOLS {IP,DECNET PHASE IV}
   ```

4. Enable the circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

9.10 CHDLC Routing Circuits

CHDLC routing circuits are type PPP circuits running over the CHDLC data link. CHDLC routing circuits have many of the same conditions attached to them as PPP routing circuits. However, they differ slightly in the way network protocols are used.

- Refer to Section 9.10.1 for information about the way network protocols are used by CHDLC circuits.
- Refer to Section 9.8.2 to Section 9.8.4 for information on PPP routing circuits that also applies to CHDLC circuits.

9.10.1 Defining Protocols for CHDLC Circuits

CHDLC routing circuits are protocol specific. This means that you must define the network protocols that will be routed over a circuit. This is described in Section 9.11.

The protocols you define depend on the configuration of your network and the type of data routed over the circuit. You can specify one or more of the protocols listed in Section 9.6.2.

9.10.2 Specifying Only One Protocol

You cannot set up the CHDLC routing circuit to run NetWare IPX only. If you specify NETWARE IPX, you must also specify one of the other protocols listed in Section 9.10.1.

However, you can set up the CHDLC routing circuit to run only one of the other network protocols: IP, ISO 8473 or DECnet Phase IV.

9.11 Adding a CHDLC Routing Circuit

9.11.1 Introduction

This section describes how to create a CHDLC routing circuit. Before you begin, you must create a CHDLC data link by following the procedures in Section 8.19.

9.11.2 Procedure

Create a new CHDLC routing circuit as follows:

1. Create a routing circuit, specifying that it is of type PPP:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name TYPE PPP
   ```

2. Specify the link that this circuit is to use as its data link:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> DATA LINK ENTITY CHDLC LINK link-name
   ```

3. Specify the network protocols to run over the circuit. If you do not specify a protocol, IP and ISO8473 are set up on the circuit automatically. Note that the remote system must support the protocols that you select.

   Enter the following command:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> NETWORK PROTOCOLS {protocol1, protocol2, protocol3}
   ```

   where *protocol1*, *protocol2*, and *protocol3* are the protocols that you require. You can specify one or more of the following:

   - IP
   - ISO8473
   - DECNET PHASE IV
   - NETWARE IPX

   For example:

   ```
   NCL> SET ROUTING CIRCUIT W622-4-1 NETWORK PROTOCOLS {IP, DECNET PHASE IV}
   ```

4. Enable the circuit:

```
NCL> ENABLE ROUTING CIRCUIT circuit-name
```

9.12 Adding an SMDS Routing Circuit

9.12.1 Introduction

This section describes how to create an SMDS routing circuit. Before you do this, you must create an SMDS STATION entity by following the procedures in Section 8.22.

9.12.2 Procedure

To create a new SMDS routing circuit, follow these steps:

1. Create a routing circuit, specifying that it is of type SMDS:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name TYPE SMDS
   ```

   where *circuit-name* is the name of the new SMDS circuit.

2. Specify the SMDS STATION entity that this circuit will use as its data link. Allocate the cost of the circuit and the frequency of the IS–IS hello timer:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> DATA LINK ENTITY SMDS STATION station-name -
   _NCL> L2 COST l2-cost -
   _NCL> ISIS HELLO TIMER hello_timer
   ```

   where:

   station-name is the name of the SMDS station.

   l2-cost is the Level 2 routing circuit cost for this circuit, in the range 1–63. The default value is 20.

   hello-timer is the frequency of the IS-IS hello timer messages, in seconds. The range is 1–32767, and the default value is 3.

3. Specify the routing circuit protocol parameters required for SMDS use:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> NETWORK PROTOCOLS {ip,iso8473}, -
   _NCL> MANUAL L2ONLY MODE l2-state, -
   _NCL> MAXIMUM ENDSYSTEM ADJACENCIES max-adj, -
   _NCL> ENABLE PHASEIV ADDRESS FALSE, -
   _NCL> MANUAL DATA LINK SDU SIZE sdu-size
   ```

where:

l2-state is TRUE if there are no end nodes.

max-adj is 0 (zero) if there are no end nodes on this circuit.

sdu-size is the size allocated to each data link SDU. The default value is 4492.

Note that you must give the attribute ENABLE PHASEIV ADDRESS the value FALSE. The value TRUE is not supported for SMDS routing circuits.

4. Enable the circuit:

```
NCL> ENABLE ROUTING CIRCUIT circuit-name
```

9.13 X.25 Routing Circuits

9.13.1 Introduction

X.25 routing circuits use X.25 virtual or permanent circuits to exchange packets. You can use X.25 routing circuits to connect remote nodes across PSDNs (Packet Switching Data Networks).

9.13.2 Types of X.25 Routing Circuit

There are two main types of X.25 routing circuit, as described in Table 9–1.

Table 9–1 Types of X.25 Routing Circuit

<table>
<tr><th>Type</th><th>Description</th></tr>
<tr><td>Dynamically Assigned</td><td>An X.25 virtual circuit is only created when there is traffic to send or receive on the X.25 DA circuit.
Adaptive routing information is not exchanged across these circuits.
Only Level 2 routers can have X.25 DA circuits.</td></tr>
<tr><td>Static</td><td>Static X.25 circuits form permanent connections between two DECnet routers.
Routing information is exchanged across these circuits.
There are three types of X.25 static circuits:
<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>Outgoing</td><td>The DECNIS initiates the X.25 call to establish the connection.</td></tr>
<tr><td>Incoming</td><td>The DECNIS accepts the X.25 call from the remote DECnet router.</td></tr>
<tr><td>Permanent</td><td>The DECNIS is connected to the remote DECnet router by means of a PVC (Permanent Virtual Circuit).</td></tr>
</table></td></tr>
</table>

9.14 Templates for X.25 Routing Circuits

9.14.1 Template Function

An X.25 routing circuit uses a template to set the X.25 call characteristics for outgoing calls.

Section 9.15 explains how to create templates, and specify the call characteristics that they will set.

9.14.2 Which X.25 Routing Circuits Require Templates

The following types of X.25 routing circuit require templates:

- X.25 Dynamically Assigned
- X.25 Static Outgoing
- X.25 Static Incoming (optional)

9.14.3 Template Characteristics

Table 9–2 shows the template characteristics required for each type of X.25 routing circuit.

Table 9–2 Template Characteristics for X.25 Routing Circuits

Circuit Type	Template Characteristic	(R)equired or (O)ptional	Value
Dynamically Assigned	DTE CLASS	R	The name of the DTE class on the DECNIS that contains the DTE(s) to be used to make calls on this circuit.
	CALL DATA	R	For DECnet/OSI templates, the first octet should be 81, for example, %X81FF11. For IP templates, the first octet should be CC, for example, %XCCFF11.
Static Outgoing	DTE CLASS	R	As above.
	CALL DATA	R	Although any value can be used, provided it is agreed with the remote system, Digital recommends that the first 14 octets are FF0000004445436E65742D444C4D, for example, %XFF0000004445436E65742D444C4D. By default, Digital routers (both Phase IV and DECnet/OSI) recognize calls with this call data as calls on static X.25 routing circuits.
	DESTINATION DTE ADDRESS	R	DTE address of the system to which the routing circuit is connected.
Static Incoming	PACKET SIZE	O	Used in call negotiation: preferred size of packet used over this circuit.
	WINDOW SIZE	O	Used in call negotiation: preferred size of window used over this circuit.

9.15 Creating and Deleting X.25 Templates

This section describes how to create a template and specify the characteristics of the outgoing calls that it will set.

9.15.1 Procedure for Creating Templates

Create a template as follows:

1. Ensure that the X25 ACCESS entity exists:

   ```
   NCL> CREATE X25 ACCESS
   ```

2. Create a TEMPLATE entity:

   ```
   NCL> CREATE X25 ACCESS TEMPLATE template-name
   ```

3. Set the template characteristics that you require:

   ```
   NCL> SET X25 ACCESS TEMPLATE template-name characteristic
   ```

   The NCL online help lists all the characteristics of the TEMPLATE entity.

9.15.2 Procedure for Deleting Templates

Delete an X.25 template as follows:

1. Ensure that no X.25 routing circuit is associated with the template to be deleted.

2. Delete the TEMPLATE entity as follows:

   ```
   NCL> DELETE X25 ACCESS TEMPLATE template-name
   ```

9.16 Filters for X.25 Routing Circuits

An X.25 routing circuit uses a filter to select incoming calls that are intended for that circuit. Refer to Section 18.5 for a general description of the operation of X.25 filters, and how to create and manage them.

9.16.1 Which X.25 Routing Circuits Require Filters

The following types of X.25 routing circuit require filters:

- X.25 Dynamically Assigned
- X.25 Static Incoming

9.16.2 Filter Characteristics

In general, a filter associated with a particular X.25 routing circuit needs to match the call characteristics of calls from the remote system to which the routing circuit is attached.

SECURITY FILTER Characteristic

You must specify the security filter to be used by the filter:

```
NCL> SET X25 ACCESS FILTER filter-name SECURITY FILTER RX25_DEFAULT
```

The security filter RX25_DEFAULT is open, that is, it allows all calls from all remote DTEs. See Section 20.3 if you want to use X.25 security to restrict incoming calls on an X.25 routing circuit.

X.25 DA Circuits Used for Both IP and DECnet/OSI Traffic

Note that you will need to create two filters for each of these circuits, to match the two different sets of call user data expected.

- For DECnet/OSI traffic, a filter should match on call user data that has a value of 81 in the first octet:

  ```
  NCL> SET X25 ACCESS FILTER circ-1-osi CALL DATA MASK %XFF, -
  _NCL> CALL DATA VALUE %X81
  ```

- For IP traffic, a filter should match on call user data that has a value of CC in the first octet:

  ```
  NCL> SET X25 ACCESS FILTER circ-1-ip CALL DATA MASK %XFF, -
  _NCL> CALL DATA VALUE %XCC
  ```

9.17 Adding an X.25 Dynamically Assigned Routing Circuit

9.17.1 Introduction

This section describes how to create an X.25 Dynamically Assigned routing circuit. It assumes that a DTE already exists to send and receive the X.25 traffic. Refer to Section 18.2.2 for instructions on creating the entities required.

9.17.2 Routing Level

The DECNIS must be configured as a Level 2 router if it has any X.25 Dynamically Assigned circuits.

9.17.3 Procedure

Create an X.25 Dynamically Assigned routing circuit as follows:

1. Set up a filter for the circuit, to determine how incoming calls are handled: see Section 18.5.2.

   Note that if an X.25 Dynamically Assigned circuit is to be used for both DECnet/OSI and IP data, you must set up a filter for each type of data. This is because the call user data is different for DECnet/OSI and IP routing.

2. Set up a template for outgoing calls: see Section 9.15. The template specifies the DTE class to be used in making calls, and puts the correct call user data in the call packet.

   Note that if an X.25 Dynamically Assigned circuit is to be used for both DECnet/OSI and IP routing, you must set up a template for each type of routing.

3. Create a routing circuit, specifying the type:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name TYPE X25 DA
   ```

4. Specify the data link that this circuit will use. This is the X.25 Access module.

   ```
   NCL> SET ROUTING CIRCUIT circuit-name DATA LINK ENTITY X25 ACCESS
   ```

5. Associate the circuit with the filter(s) created in step 1:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> X25 FILTERS {filter-name1, filter-name2, ...}
   ```

6. Associate the circuit with the template(s) created in step 2.

   Use one or both of the following commands:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name TEMPLATE template-name
   ```

   ```
   NCL> SET ROUTING CIRCUIT circuit-name IP TEMPLATE IP-template-name
   ```

   where *template-name* is the name of a DECnet/OSI template, and *IP-template-name* is the name of an IP template: see Section 9.15.1.

7. Set up the manual routing required for this circuit: see Section 16.15 and Section 10.14. (All routing over X25 DA circuits is manual.)

8. Enable the circuit:

   ```
   NCL>  ENABLE ROUTING CIRCUIT circuit-name
   ```

9.18 Adding an X.25 Static Outgoing Routing Circuit

9.18.1 Introduction

This section describes how to create an X.25 Static Outgoing routing circuit. It assumes that a DTE already exists to send and receive the X.25 traffic.

9.18.2 Procedure

Create an X.25 Static Outgoing routing circuit as follows:

1. Set up a template for outgoing calls (see Section 9.15). The template specifies the DTE class to be used in making calls, and puts the correct call user data in the call packet. It also specifies the remote DTE address.
2. Create a routing circuit, specifying the type:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name TYPE X25 STATIC OUTGOING
   ```

3. Specify the data link that this circuit will use. This is the X.25 Access module.

   ```
   NCL> SET ROUTING CIRCUIT circuit-name DATA LINK ENTITY X25 ACCESS
   ```

4. Associate the circuit with the template created in step 1:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name TEMPLATE template-name
   ```

5. Enable the circuit:

   ```
   NCL>  ENABLE ROUTING CIRCUIT circuit-name
   ```

9.19 Adding an X.25 Static Incoming Routing Circuit

9.19.1 Introduction

This section describes how to create an X.25 Static Incoming routing circuit. It assumes that a DTE already exists to send and receive the X.25 traffic.

9.19.2 Procedure

Create an X.25 Static Incoming routing circuit as follows:

1. Set up a filter for the circuit, to determine how incoming calls are handled.
2. If required, set up a template for setting parameters during call negotiation: see Section 9.15.
3. Create a routing circuit, specifying the type:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name TYPE X25 STATIC INCOMING
   ```

4. Specify the data link that this circuit will use. This is the X.25 Access module.

   ```
   NCL> SET ROUTING CIRCUIT circuit-name DATA LINK ENTITY X25 ACCESS
   ```

5. Associate the circuit with the filter(s) created in step 1:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> X25 FILTERS {filter-name1, filter-name2, ...}
   ```

6. Associate the circuit with the template(s) created in step 2 (if any):

   ```
   NCL> SET ROUTING CIRCUIT circuit-name TEMPLATE template-name
   ```

7. Enable the circuit:

   ```
   NCL>  ENABLE ROUTING CIRCUIT circuit-name
   ```

9.20 Adding an X.25 Permanent Routing Circuit

9.20.1 Introduction

This section describes how to create an X.25 Permanent routing circuit. It assumes that a DTE already exists to send and receive the X.25 traffic.

9.20.2 Procedure

Create an X.25 Permanent routing circuit as follows:

1. Set up a PVC on the DTE to be used for the routing circuit: see Section 18.6.2.
2. Create a routing circuit, specifying the type:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name TYPE X25 PERMANENT
   ```

3. Specify the data link that this circuit will use. This is the X.25 Access module.

   ```
   NCL> SET ROUTING CIRCUIT circuit-name DATA LINK ENTITY X25 ACCESS
   ```

4. Specify the name of the PVC created in step 1:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name TEMPLATE PVC-name
   ```

5. Enable the circuit:

   ```
   NCL>  ENABLE ROUTING CIRCUIT circuit-name
   ```

9.21 Changing Routing Circuit Characteristics

This section describes some circuit characteristics that you may wish to change from their default values. Each of the subsections gives the following information:

- The type of routing circuit to which the characteristic applies.
- A description of the characteristic.
- How to change the characteristic.

9.21.1 Changing the Phase IV Routing Vector Timer on PPP, CHDLC and ATM Permanent Circuits

Routing circuit type: PPP, CHDLC, ATM Permanent)

9.21.1.1 Introduction

The Phase IV Routing Vector Timer sets the time between each routing vector packet that is transmitted over PPP, CHDLC and ATM Permanent circuits.

The Phase IV Routing Vector Timer is useful if, for example, you want to increase the frequency of routing vector updates on circuits of these types that have a high error rate.

If you are using CHDLC connections to a DECbrouter 90 you cannot use the SLARP TRANSMISSION TIMER, so you may find this timer useful to monitor the circuit.

The timer is set to 30 seconds by default.

9.21.1.2 Procedure

Enter the following command:

```
NCL> SET ROUTING CIRCUIT circuit-name PHASE IV ROUTING VECTOR TIMER time
```

where *time* represents a value in seconds between 1 and 65,535. For example:

```
NCL> SET ROUTING CIRCUIT W622-4-1 PHASE IV ROUTING VECTOR TIMER 15
```

9.21.2 Changing the Maximum Data Link Message Size

9.21.2.1 Description

This section describes how to specify the largest data link message to be sent over a circuit. You may want to set this to a smaller value than the default (1492) if the circuit connects to an end system.

9.21.2.2 FDDI and ATM Permanent Routing Circuits

For FDDI and ATM Permanent routing circuits, you may want to increase the MANUAL DATA LINK SDU SIZE from its default value of 1492. However, note the following:

- You can increase this value up to 4481 for FDDI or up to 4492 for ATM, but other systems in the network may not support such large packet sizes.
- If you need to ensure operability with DECnet Phase IV systems, then keep the default value of 1492.

9.21.2.3 Procedure

Set the maximum data link message size for a routing circuit as follows:

1. Disable the routing circuit:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit-name
   ```

2. Set the MANUAL DATA LINK SDU SIZE for the circuit:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> MANUAL DATA LINK SDU SIZE n
   ```

   where *n* is a decimal number between 128 and 65,535.

3. Enable the routing circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

9.21.3 Changing the Originating Queue Limit

Routing circuit type: CSMA/CD, HDLC, DDCMP, X.25, FDDI

9.21.3.1 Description

The originating queue limit specifies the number of data PDUs originated by the DECNIS that can be on this circuit's transmit queue. Use the minimum value required to prevent the data link from idling.

Only change the value of this characteristic if you are using X.25 gateway or relay functions on the DECNIS.

Recommended value

Normally, if you are using X.25 gateway or relay functions on the DECNIS, you should set this value to 4. If you are not using X.25 gateway or relay functions, leave the value as 2 (the default).

9.21.3.2 Procedure

Set the originating queue limit for a routing circuit as follows:

```
NCL> SET ROUTING CIRCUIT circuit-name ORIGINATING QUEUE LIMIT n
```

where *n* is a decimal number between 1 and infinity.

9.21.4 Changing the Recall Timer

Routing circuit type: PPP, X25 DA, X25 STATIC OUTGOING

9.21.4.1 Description

On PPP routing circuits, the recall timer specifies the time that must elapse between a failed initialization attempt and a retry.

On X25 DA and X25 STATIC OUTGOING routing circuits, the recall timer specifies the time that must elapse between an outgoing call on the circuit failing and a retry.

Guideline

Specify a value that is high enough to allow subsequent retries a reasonable chance of success. For example, setting a value of 0 means that there is no delay (0 seconds) between a failure and the next retry.

On X25 STATIC OUTGOING routing circuits, setting the value 0 means that the circuit will use up all of its call attempts (see Section 9.21.5) before succeeding in making the call.

9.21.4.2 Procedure

Set the recall timer for a routing circuit as follows:

```
NCL> SET ROUTING CIRCUIT circuit-name RECALL TIMER n
```

where *n* is a decimal number between 0 and 65,535.

9.21.5 Changing the Maximum Call Attempts

Routing circuit type: X25 STATIC OUTGOING

9.21.5.1 Description

The maximum call attempts for a routing circuit is the number of successive call failures allowed on the circuit before it is regarded as halted.

9.21.5.2 Procedure

Set the maximum call attempts for a routing circuit as follows:

```
NCL> SET ROUTING CIRCUIT circuit-name MAXIMUM CALL ATTEMPTS n
```

where *n* is a decimal number between 0 and 255. A value of 0 means that there is no limit on the number of retries.

9.21.6 Changing the Maximum SVC Adjacencies

Routing circuit type: X25 DA

9.21.6.1 Description

This specifies the maximum number of adjacencies allowed on this circuit; the number of SVCs plus the number of dormant adjacencies on the circuit cannot exceed the value of this characteristic.

Note that you can increase this value without disabling the circuit.

9.21.6.2 Procedure

Set the maximum SVC adjacencies for a routing circuit as follows:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name
NCL> SET ROUTING CIRCUIT circuit-name MAXIMUM SVC ADJACENCIES n
NCL> ENABLE ROUTING CIRCUIT circuit-name
```

where *n* is a decimal number between 1 and 65,535.

9.21.7 Changing the Idle Timer

Routing circuit type: X25 DA

9.21.7.1 Description

The idle timer indicates the time, in seconds, for which data traffic on the circuit must be absent before the call is cleared.

9.21.7.2 Procedure

Set the idle timer for a routing circuit as follows:

```
NCL> SET ROUTING CIRCUIT circuit-name IDLE TIMER n
```

where *n* is a decimal number between 1 and 65,535.

9.21.8 Changing the Initial Minimum Timer

Routing circuit type: X25 DA

9.21.8.1 Description

This indicates the minimum time, in seconds, that a call on a circuit will remain connected, irrespective of traffic.

9.21.8.2 Procedure

Set the initial minimum timer for a routing circuit as follows:

```
NCL> SET ROUTING CIRCUIT circuit-name INITIAL MINIMUM TIMER n
```

where *n* is a decimal number between 1 and 65,535.

9.22 Using NCL to Monitor a Routing Circuit

You can use the following commands to check how a routing circuit is functioning:

- To check if a circuit is enabled:

  ```
  NCL> SHOW ROUTING CIRCUIT circuit-name STATUS
  ```

- To check how a circuit is being used:

  ```
  NCL> SHOW ROUTING CIRCUIT circuit-name ALL COUNTERS
  ```

  See the NCL online help for a description of the counters displayed.

9.23 Deleting a Routing Circuit

9.23.1 Introduction

Delete a routing circuit if you want to prevent permanently the DECNIS from communicating with the system(s) connected by that circuit. Use the DISABLE command if you want to prevent temporarily the DECNIS from communicating with the system(s).

9.23.2 Deleting Non-X.25 Routing Circuits

Delete a CSMA/CD, FDDI, HDLC, DDCMP, PPP, ATM Permanent, CHDLC or SMDS routing circuit as follows:

1. Disable and delete any reachable addresses or IP reachable addresses of the ROUTING CIRCUIT entity:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name REACHABLE ADDRESS *
NCL> DISABLE ROUTING CIRCUIT circuit-name IP REACHABLE ADDRESS *

NCL> DELETE ROUTING CIRCUIT circuit-name REACHABLE ADDRESS *
NCL> DELETE ROUTING CIRCUIT circuit-name IP REACHABLE ADDRESS *
```

2. Disable the routing circuit:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit-name
   ```

3. Delete the routing circuit:

   ```
   NCL> DELETE ROUTING CIRCUIT circuit-name
   ```

9.23.3 Deleting X.25 Routing Circuits

Delete an X.25 routing circuit as follows:

1. Follow the steps in Section 9.23.2.
2. Delete the templates associated with this routing circuit, unless they are being used by another routing circuit: see Section 9.15.2.
3. Delete any filters associated with this routing circuit, unless they will be used by another routing circuit: see Section 18.5.3.
4. Delete the DTE, if it will not be used for any other functions: see Section 18.2.9.

9.24 Changing the Routing Level of the DECNIS

9.24.1 Description of Routing Level

You can configure the DECNIS to be either a Level 1 or a Level 2 router:

- A **Level 1 router** forwards packets only to systems within its own area.
- A **Level 2 router** can behave like a Level 1 router, but can also forward packets to other areas of the network, and to other routing domains.

9.24.2 Cannot Change Routing Level of a Running System

To change the routing level of your DECNIS, you must run the DECNIS configurator (text-based or Windows), create a new configuration file and reboot the system. You cannot change the routing level of a running system.

9.24.3 Procedure: DECNIS Text-based Configurator

To change the routing level of the DECNIS, supply the following information to the DECNIS configurator (text-based):

	Screen	Action
1	Sections Menu	Select Routing.
2	Routing Options Menu	Select Modify Routing Information.
3	Routing	Select Level 1 or Level 2, and enter routing algorithm and address information as necessary.

9.24.4 Procedure: clearVISN DECNIS Configurator

To change the routing level of the DECNIS, follow these steps in the clearVISN DECNIS configurator:

1. Click on **DNA IV/V + OSI** on the main navigation window.
2. On the General tab page, select the routing level you want under **Mode**.
3. Click **OK**.

9.24.5 Effect of Changing the Routing Level

In the DECNIS text-based configurator, changing the routing level from Level 2 to Level 1 will cause the configurator to delete X.25 dynamically assigned circuits, IP and OSI reachable addresses, and EGP information. Changing the routing level from Level 1 to Level 2 will cause the routing circuit characteristics that apply only to Level 2 routers to be set to their default values. You can change these as required.

In the clearVISN DECNIS configurator, changing the routing level from Level 2 to Level 1 will cause the IP and OSI reachable addresses to become inoperative.

9.25 Preventing Routing Circuits from Routing Level 1 Traffic

By default, all the routing circuits of a Level 2 router can route both Level 1 and Level 2 traffic.

This section describes how to prevent a routing circuit on a DECNIS, running the link state routing algorithm at Level 2, from routing Level 1 traffic.

9.25.1 Procedure

To prevent a routing circuit from routing Level 1 traffic, follow these steps:

1. Disable the routing circuit:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit-name
   ```

2. Set the MANUAL L2ONLY MODE characteristic to TRUE:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name MANUAL L2ONLY MODE true
   ```

3. Enable the routing circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

9.25.2 Consequences of Changing from TRUE to FALSE

If you change the MANUAL L2ONLY MODE from TRUE to FALSE, set the L1 COST and L1 ROUTER PRIORITY for the circuit, if necessary (see Section 16.11 and Section 16.13, respectively).

9.26 Routing Security

9.26.1 Introduction

This section describes how to prevent an unauthorized adjacent system from setting up a circuit to connect with your DECNIS. It applies only to HDLC, DDCMP and X.25 STATIC routing circuits. Section 20.3 describes how to use X.25 security for other types of X.25 routing circuit.

9.26.2 Routing Security Mechanism

You secure the routing circuits on your DECNIS by creating verifiers (passwords). Systems that want to contact your system must supply the verifier that you set.

9.26.3 Circuit and System Routing Security

There are two methods of setting verifiers:

- On a **circuit-by-circuit** basis. You assign a verifier to each circuit. Any remote system wanting to use that circuit must supply the circuit verifier.
- On a **system-by-system** basis. You assign one or more verifiers to each system to which your DECNIS is connected. Any system that wants to communicate with your system, on any HDLC, DDCMP or X.25 STATIC routing circuits, must supply the appropriate verifier.

  This method is useful if there are several systems that can reach your DECNIS on the same circuit; for example, an X.25 or a dialup circuit.

Note that you can use circuit-level security for some circuits on the DECNIS, and leave other circuits to be protected at the system level.

9.26.4 How Routing Security Works

When an adjacent system tries to set up a circuit to the DECNIS, the checks shown in Figure 9–5 are made.

9.27 Setting Up Circuit-Level Routing Security

9.27.1 Introduction

This section describes how to set up a verifier (password) for a synchronous routing circuit. Any system that tries to use this routing circuit to communicate with your system must supply this verifier, irrespective of any system-level security on the DECNIS.

Circuit Types

You can only set up a verifier for a routing circuit if the circuit type is HDLC, DDCMP, or X.25 STATIC INCOMING.

9.27.2 Procedure

Set up routing security on a routing circuit as follows:

1. Set the verifier on the circuit as follows:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name RECEIVE VERIFIER verifier-value
   ```

   where *verifier-value* is an octet string, which takes the form %x*n*. *n* is an even number of hexadecimal digits; an example verifier value is %x12AB.

   If the adjacent system is a DECnet Phase IV system, it specifies the verifier as a text string. In this case, you convert each letter in the text string to its ASCII value in hexadecimal, and enter this as the value of *n*.

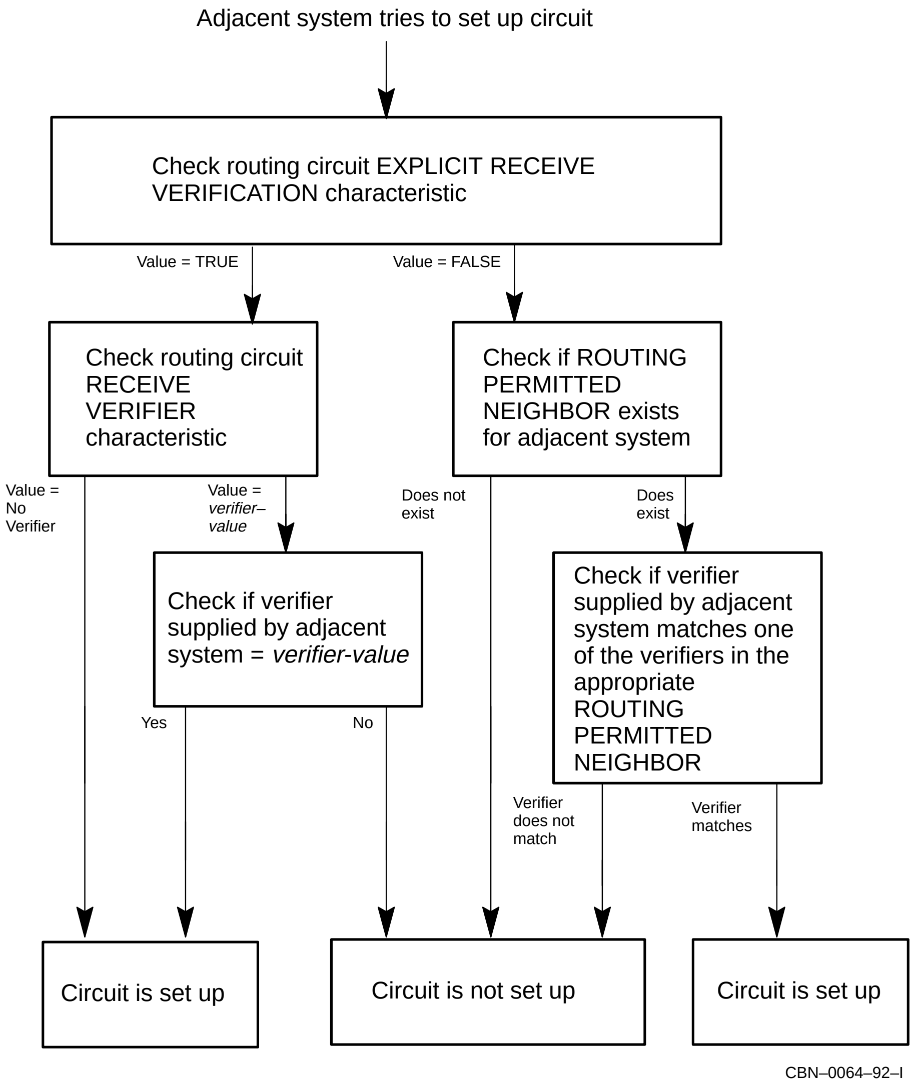
   For example: the text string SECRET becomes %x534543524554.

2. Set the circuit to check for this verifier when the circuit is initialized:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> EXPLICIT RECEIVE VERIFICATION TRUE
   ```

Although you can issue these commands while the circuit is enabled, they will have no effect until the circuit is next initialized. If you want them to take effect immediately, disable and enable the circuit after issuing them.

Figure 9–5 Security Checks Made by Routing


Adjacent system tries to set up circuit
Check routing circuit EXPLICIT RECEIVE VERIFICATION characteristic
Value = TRUE
Value = FALSE
Check routing circuit RECEIVE VERIFIER characteristic
Check if ROUTING PERMITTED NEIGHBOR exists for adjacent system
Value = No Verifier
Value = *verifier–value*
Does not exist
Does exist
Check if verifier supplied by adjacent system = *verifier-value*
Check if verifier supplied by adjacent system matches one of the verifiers in the appropriate ROUTING PERMITTED NEIGHBOR
Yes
No
Verifier does not match
Verifier matches
Circuit is set up
Circuit is not set up
Circuit is set up
CBN–0064–92–I

9.28 Setting Up System-Level Routing Security

9.28.1 Introduction

This section describes how to set up system-level routing security on the DECNIS.

It also describes how to disable circuit-level security for those circuits that will be protected at the system level.

9.28.2 Procedure

Set up routing security on the DECNIS as follows:

1. Disable inbound circuit-level security on those circuits to which system-level security will apply. Issue the following command for each circuit:

   ```
   NCL>  SET ROUTING CIRCUIT circuit-name -
   _NCL> EXPLICIT RECEIVE VERIFICATION FALSE
   ```

   If you do not want to use any circuit-level security on the DECNIS, issue the following command:

   ```
   NCL>  SET ROUTING CIRCUIT * -
   _NCL> EXPLICIT RECEIVE VERIFICATION FALSE
   ```

   Although you can issue these commands while the circuit is enabled, they will have no effect until the circuit is next initialized.

2. Create a PERMITTED NEIGHBOR entity for each system from which you want to accept connections:

   ```
   NCL> CREATE ROUTING PERMITTED NEIGHBOR neighbor-id ID remote-node-id
   ```

   where:

   neighbor-id identifies the PERMITTED NEIGHBOR entity. The name you use should be based on the name of the remote system concerned.

   remote-node-id specifies the system ID of the remote system that will have to supply a verifier. Section 16.2 describes system IDs.

3. Set the verifier that this remote system must use:

   ```
   NCL> SET ROUTING PERMITTED NEIGHBOR -
   _NCL> neighbor-id VERIFIER verifier-value
   ```

   where *verifier-value* is an octet string, which takes the form %x*n*. *n* is an even number of hexadecimal digits; an example verifier value is %x12AB.

   If the remote system is a DECnet Phase IV system, it specifies the verifier as a text string. In this case, you convert each letter in the text string to its hexadecimal value in ASCII, and enter this as the value of *n*.

   For example: the text string SECRET becomes %x534543524554.

9.29 Configuring the DECNIS to Send Verifiers

9.29.1 Introduction

If an adjacent system has set up either form of security on the routing circuit(s) that connect to your system, your system must supply a verifier when it attempts to communicate with the adjacent system.

This section describes how to configure a routing circuit on the DECNIS to send a verifier to an adjacent system.

You can also configure verifiers in either of the DECNIS configurators.

9.29.2 Procedure

Set the verifier that the routing circuit will send to the adjacent system using the following command:

```
NCL> SET ROUTING CIRCUIT circuit-id TRANSMIT VERIFIER verifier-value
```

where *verifier-value* is an octet string, which takes the form %x*n*. *n* is an even number of hexadecimal digits; an example verifier value is %x12AB.

9.30 Setting Up Backup Circuits

9.30.1 Introduction

The Backup Circuit facility enables the DECNIS to detect the failure of a routing circuit on a serial line, and establish an alternative circuit for the duration of the failure.

To use this facility, you set up **Supervisor Group** entities. Each Supervisor Group associates a **Primary circuit** and a **Secondary circuit**.

- The Primary circuit is the circuit that is monitored to detect failure.
- The Secondary circuit is the circuit that is established if the Primary circuit fails.

  The Secondary circuit is disabled while the Primary circuit is working; it only comes into operation when the Primary circuit fails.

9.30.2 Types of Routing Circuit Required

Note the following:

- Primary and Secondary circuits must be Routing Circuits on serial lines.
- You can only use certain types of routing circuit as Primary and Secondary circuits, as shown in Table 9–3.

Table 9–3 Types of Routing Circuits Supported

Primary circuits can use...	Secondary circuits can use...
HDLC circuits	HDLC circuits
DDCMP circuits	DDCMP circuits
PPP circuits	PPP circuits
	CHDLC circuits
	X.25 Static circuits
	X.25 DA circuits

9.30.3 Before You Specify a Primary Circuit

Before you specify a Primary circuit, make the following checks:

Step	Check
1.	Check that the routing circuit to be used as a Primary circuit has been created, together with its associated Modem Connect Line and data link entities.
2.	Check that the routing circuit and its associated Modem Connect Line and data link entities are enabled.

9.30.4 Before You Specify a Secondary Circuit

Before you specify a Secondary circuit, make the following checks:

Step	Check
1.	Check that the routing circuit to be used as a Secondary circuit has been created, together with its associated Modem Connect Line and the data link entities.

Step	Check
2.	For **PPP**, **HDLC**, **CHDLC** or **DDCMP** circuits, do the following: • Check that the Modem Connect Line and its associated Routing Circuit are **disabled**. • Check that the data link entities associated with the Routing Circuit are **enabled**.
3.	For **X.25 Routing** circuits, do the following: • Check that the routing circuit is **disabled**. • Check that the Modem Connect Line and the data link entities associated with the routing circuit are **enabled**.

Refer to Section 9.30.9 for more information.

9.30.5 Setting Up a Supervisor Group

To set up a Supervisor Group, follow these steps:

1. Ensure that the SUPERVISOR module exists. Enter the following:

   ```
   NCL> CREATE SUPERVISOR
   ```

   Result: A message will tell you if the SUPERVISOR module already exists; otherwise, it will be created.

2. Create a Supervisor Group:

   ```
   NCL> CREATE SUPERVISOR GROUP group-name FUNCTION BACKUP
   ```

3. Specify the Primary circuit:

   ```
   NCL> SET SUPERVISOR GROUP group-name -
   _NCL> PRIMARIES {ROUTING CIRCUIT pri-circ-name}
   ```

   where *pri-circ-name* is the routing circuit specified as the Primary circuit.

4. Specify the Secondary circuit:

   ```
   NCL> SET SUPERVISOR GROUP group-name -
   _NCL> SECONDARY ROUTING CIRCUIT sec-circ-name
   ```

   where *sec-circ-name* is the routing circuit specified as the Secondary circuit.

5. Ensure that the routing circuit used as the Primary circuit, together with its associated Modem Connect Line and data link entities, are enabled. For example, for HDLC circuits, you would enter:

   ```
   NCL> ENABLE MODEM CONNECT LINE line-name
   NCL> ENABLE HDLC LINK link-name
   NCL> ENABLE HDLC LINK link-name LOGICAL STATION station-name
   NCL> ENABLE ROUTING CIRCUIT pri-circ-name
   ```

   For more information about enabling data links, refer to Chapter 8.

6. Enable the Supervisor Group:

   ```
   NCL> ENABLE SUPERVISOR GROUP group-name
   ```

9.30.6 Setting Timers

The following table shows the timers that control the establishment and disabling of Secondary circuits.

Timer	Description	Default
INVOKE TIMER	The number of seconds to wait after a Primary circuit fails before the Secondary circuit is established.	5 seconds
REVOKE TIMER	The number of seconds to wait after a Primary circuit recovers before the Secondary circuit is disabled.	30 seconds

You may need to adjust the INVOKE TIMER if there are predictable transient delays on the network. For example, if there are frequent failures lasting no more than 10 seconds, you could increase the INVOKE timer to 10.

You may need to adjust the REVOKE TIMER if you do not want the Secondary circuit to be disabled as soon as the Primary circuit recovers, in case it fails again.

Procedure

To set the INVOKE timer, enter the following command:

```
NCL> SET SUPERVISOR GROUP group-name INVOKE TIMER n
```

where *n* is a decimal number between 1 and 65,535.

To set the REVOKE timer, enter the following command:

```
NCL> SET SUPERVISOR GROUP group-name REVOKE TIMER n
```

where *n* is a decimal number between 1 and 65,535.

9.30.7 Chaining Circuits

You can form a chain of backup circuits to take over from each other. You do this by making the Secondary circuit of one Group the Primary circuit of another.

9.30.7.1 Example

You have done the following:

- Created three DECNIS Routing Circuits:

  HDLC routing circuit **hdlc-main**
  HDLC routing circuit **hdlc-backup**
  X.25 Static routing circuit **x25-backup**

- Created the SUPERVISOR module.

You want to achieve the following:

- **hdlc-main** should be active whenever possible.
- **hdlc-backup** will be the backup circuit when **hdlc-main** fails.
- **x25-backup** will be the backup circuit when **hdlc-backup** fails.

Procedure

To set up the required backup circuits, you create two Supervisor Groups. Follow these steps:

1. Create two Supervisor Groups:

```
NCL> CREATE SUPERVISOR GROUP main_group FUNCTION BACKUP
NCL> CREATE SUPERVISOR GROUP backup_group FUNCTION BACKUP
```

2. Specify the Primary circuit in each group:

```
NCL> SET SUPERVISOR GROUP main_group -
_NCL> PRIMARIES {ROUTING CIRCUIT hdlc_main}

NCL> SET SUPERVISOR GROUP backup_group -
_NCL> PRIMARIES {ROUTING CIRCUIT hdlc_backup}
```

3. Specify the Secondary circuit in each group:

```
NCL> SET SUPERVISOR GROUP main_group -
_NCL> SECONDARY ROUTING CIRCUIT hdlc_backup

NCL> SET SUPERVISOR GROUP backup_group -
_NCL> SECONDARY ROUTING CIRCUIT x25_backup
```

4. Enable the routing circuit identified as the Primary circuit in Supervisor Group main_group:

   ```
   NCL> ENABLE ROUTING CIRCUIT hdlc_main
   ```

5. Enable both Supervisor Groups:

   ```
   NCL> ENABLE SUPERVISOR GROUP main_group
   NCL> ENABLE SUPERVISOR GROUP backup_group
   ```

Result

If the circuit hdlc_main fails, then the circuit hdlc_backup is automatically established. If the circuit hdlc_backup fails in turn, the circuit x25_backup is automatically established.

9.30.8 Testing Secondary Circuits

This section describes how to test that a Secondary circuit will be correctly established when the Primary circuit fails.

Procedure

Issue the following command:

```
NCL> TEST SUPERVISOR GROUP group-name TIMER 10
```

where *group-name* is the name of the Supervisor Group with the Secondary circuit.

Result: If the Secondary circuit could not be enabled, an error will be displayed.

The TIMER parameter specifies the amount of time that the Secondary circuit will remain enabled. If you omit the TIMER parameter, the REVOKE TIMER value for the Supervisor Group is used instead.

9.30.9 Disabling Secondary Circuit Entities

A Secondary circuit should not be used for sending data unless the Primary circuit fails. The Secondary circuit needs to be inactive, but available for use.

You achieve this for PPP, HDLC and DDCMP Routing Circuits by disabling the Modem Connect Line, and for X.25 routing circuits by disabling the Routing Circuit. If the Primary circuit fails, the software automatically enables the required entity for the Secondary circuit.

Enabling and Disabling the Modem Connect Line

The effect of enabling and disabling the Modem Connect Line for PPP, HDLC and DDCMP Routing Circuits is to assert and deassert the DTR modem signal, causing the external equipment to establish or break the call.

Note that if you do not disable the Modem Connect Line, calls may be established when circuits are not in use.

9.30.10 Changing the Modem Connect Line

The software establishes the identity of the Modem Connect Line for the Secondary (and Primary) circuits when the Supervisor Group is enabled. If you change the association between the Secondary Routing Circuit and the Modem Connect Line subsequently, then unexpected results may occur.

9.30.11 Changing the Characteristics of Backup Circuits

You must disable a SUPERVISOR GROUP entity before you do either of the following:

- Change the characteristics of the Primary or Secondary circuits in the Supervisor Group.
- Change the routing circuit characteristics of a Primary or Secondary circuit in the Supervisor Group.

9.31 Optimizing the Routing Database Sizes on the DECNIS

9.31.1 Introduction

The DECNIS maintains a number of databases in which it stores routing information. The sizes of these databases determine the size of network that the DECNIS can support.

This section describes how to correctly size these routing databases to ensure that the DECNIS makes the most efficient use of its available memory.

9.31.2 How Routing Database Size Is Implemented on the DECNIS

The ROUTING module entity has a number of status attributes and characteristic attributes, as shown in Table 9–4.

The value of each characteristic controls the memory reserved for a particular database or group of databases.

The value of the related status attributes shows the memory actually being used by this database.

For OSPF ROUTING characteristic attributes, refer to Section 9.31.3.

Table 9–4 ROUTING Attributes Relating to Routing Database Sizes

Characteristic	Explanation	Associated Status
MAXIMUM ROUTER ADJACENCIES	Total number of adjacent Level 1 and Level 2 routers	CURRENT L1 ROUTER ADJACENCIES PEAK L1 ROUTER ADJACENCIES CURRENT L2 ROUTER ADJACENCIES PEAK L2 ROUTER ADJACENCIES
MAXIMUM ENDSYSTEM ADJACENCIES	Total number of adjacent end systems.	CURRENT ENDSYSTEM ADJACENCIES PEAK ENDSYSTEM ADJACENCIES
MAXIMUM ENDSYSTEM ADJACENCIES (circuit)	This is a ROUTING CIRCUIT characteristic that applies only to CSMA/CD and FDDI routing circuits. It specifies the maximum number of adjacent end systems (ENDNODE IDs) permitted on each routing circuit.	
MAXIMUM MANUAL ADJACENCIES	Total number of Level 1 manual adjacencies.	
MAXIMUM REACHABLE DESTINATIONS	Total number of OSI reachable addresses (Level 2 manual adjacencies).	
MAXIMUM DA ADJACENCIES	Total number of X.25 dynamically assigned circuits	CURRENT DA ADJACENCIES PEAK DA ADJACENCIES
MAXIMUM L1 ROUTING DESTINATIONS	Total number of unique addresses in the same area as the DECNIS.	CURRENT L1 ROUTING DESTINATIONS PEAK L1 ROUTING DESTINATIONS
MAXIMUM L1 ROUTERS	Total number of Level 1 routers in the same area as the DECNIS.	

(continued on next page)

Table 9–4 (Cont.) ROUTING Attributes Relating to Routing Database Sizes

Characteristic	Explanation	Associated Status
MAXIMUM L2 ROUTING DESTINATIONS	Total number of area addresses in the same routing domain as the DECNIS, plus the number of OSI reachable addresses on the DECNIS.	CURRENT L2 ROUTING DESTINATIONS PEAK L2 ROUTING DESTINATIONS
MAXIMUM L2 ROUTERS	Total number of Level 2 routers in the same routing domain as the DECNIS.	
MAXIMUM L1 LSPS	Total number of link state packets from Level 1 routers in the same area as the DECNIS that the DECNIS can store. If left at its default of 0, the DECNIS will estimate an appropriate value when it boots.	CURRENT L1 LSPS PEAK L1 LSPS ALLOCATED L1 LSPS CURRENT L1 LSP DATABASE PEAK L1 LSP DATABASE ALLOCATED L1 LSP DATABASE
MAXIMUM L2 LSPS	Total number of link state packets from Level 2 routers that the DECNIS can store. If left at its default of 0, the DECNIS will estimate an appropriate value when it boots.	CURRENT L2 LSPS PEAK L2 LSPS ALLOCATED L2 LSPS CURRENT L2 LSP DATABASE PEAK L2 LSP DATABASE ALLOCATED L2 LSP DATABASE
L1 CONNECTIVITY	A value describing the Level 1 network topology. See Section 9.31.4.	
L2 CONNECTIVITY	A value describing the Level 2 network topology. See Section 9.31.4.	
MAXIMUM IP LOCAL ADJACENCIES	Total number of IP subnets to which the DECNIS is directly connected plus the number of neighbor IP addresses set up on the DECNIS.	CURRENT IP LOCAL ADJACENCIES PEAK IP LOCAL ADJACENCIES
MAXIMUM IP REACHABLE DESTINATIONS	Total number of IP reachable addresses.	

(continued on next page)

Table 9–4 (Cont.) ROUTING Attributes Relating to Routing Database Sizes

Characteristic	Explanation	Associated Status
MAXIMUM IP L1 DESTINATIONS	Total number of unique IP addresses in the same area as the DECNIS.	CURRENT IP L1 DESTINATIONS PEAK IP L1 DESTINATIONS
MAXIMUM IP L2 DESTINATIONS	Total number of area addresses in the same routing domain as the DECNIS.	CURRENT IP L2 DESTINATIONS PEAK IP L2 DESTINATIONS
MAXIMUM IP EXTERNAL DESTINATIONS	Total number of IP destinations learnt from RIP and EGP.	CURRENT IP EXTERNAL DESTINATIONS PEAK IP EXTERNAL DESTINATIONS
IP L1 CONNECTIVITY	A value describing the Level 1 IP network topology. See Section 9.31.4.	
IP L2 CONNECTIVITY	A value describing the Level 2 IP network topology. See Section 9.31.4.	

9.31.3 OSPF Routing Characteristics

Table 9–5 shows the characteristic attributes associated with the OSPF GENERAL ROUTING CONTROL PROTOCOL.

If there is an OSPF shutdown, then you may need to look at the current and peak values of the ROUTING CONTROL PROTOCOL status attributes, and change database sizes. The DECNIS online Problem Solving manual describes how to monitor and change the required OSPF attributes.

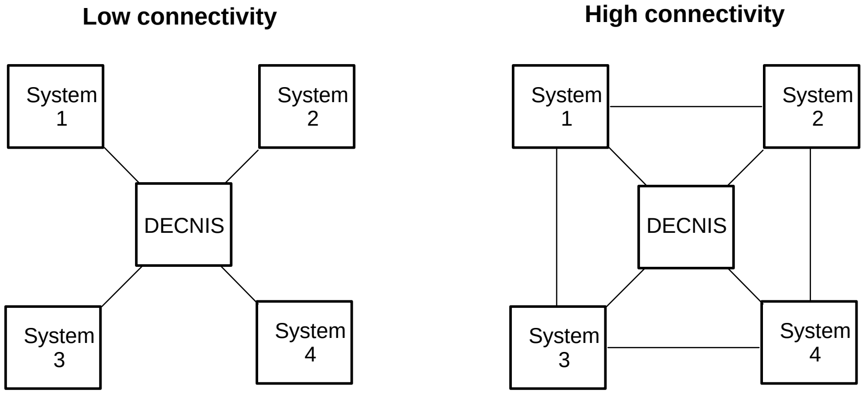
Table 9–5 OSPF General Routing Control Protocol Characteristic Attributes

Characteristic	Explanation
OSPF MAXIMUM CONNECTED AREAS	Maximum number of OSPF areas that the router can connect to directly.
OSPF AVERAGE CONNECTED ROUTERS	Average number of OSPF routers in each area that the router is connected to directly.
OSPF MAXIMUM AREA INTERFACES	Maximum number of OSPF Interfaces to a single area on any OSPF router in a connected area.
OSPF AVERAGE AREA NETWORKS	Average number of OSPF transit and stub networks in each area that the router is connected to directly.
OSPF MAXIMUM NETWORK ROUTERS	Maximum number of OSPF routers in any OSPF network in a connected area.
OSPF MAXIMUM SYSTEM NETWORKS	Maximum number of OSPF networks in the autonomous system. This is the total number of discrete area ranges defined in the autonomous system *plus* the number of IP networks in areas that have no area ranges defined *plus* the number of IP networks in the backbone area if virtual links are used.
OSPF MAXIMUM BOUNDARY ROUTERS	Maximum number of OSPF autonomous system boundary routers in the autonomous system.
OSPF MAXIMUM EXTERNAL ROUTERS	Maximum number of OSPF external routes in the autonomous system. **Note:** Per IP destination and for each forwarding address, there is a separate external route.
OSPF AVERAGE EXTERNAL CONNECTIVITY	Average number of discrete forwarding addresses provided by each boundary router for OSPF external routes.
MAXIMUM DESTINATIONS	Maximum number of destinations that this protocol can have in the routing table.
MAXIMUM ADJACENCIES	Maximum number of adjacencies that this protocol can form.

9.31.4 Connectivity: A Measure of the "Connectedness" of a Network

The connectivity of a network is a measure of how interconnected the nodes in a network are. Figure 9–6 illustrates how a simple network can be connected to give different connectivity factors.

Figure 9–6 Connectivity



Connectivity is approximately 1. DECNIS learns about each node from only one source.

Connectivity is approximately 4. DECNIS learns about each node from four sources. For example, it learns about System 1 from System 1, System 2, System 3 and System 4.

There is no simple way to calculate the connectivity factor of a realistic network. However, one method of estimating it is to calculate the average number of routing circuits per router for the network.

Characteristic Attributes for Connectivity

Note that the connectivity of a network as described by the connectivity characteristics in Table 9–4 is one-tenth the value of the attribute.

Example: If the Level 1 IP network connectivity of your network is approximately 4, then set IP L1 CONNECTIVITY to 40.

9.31.5 Dynamically Allocated DECNIS Memory

The DECNIS allocates memory for the databases described above when the ROUTING entity is enabled.

In addition to this memory, the DECNIS allocates memory dynamically for X.25 functions, and NetWare IPX and AppleTalk routing. The amount of DECNIS memory required for these functions is shown in Table 9–6.

9.31.6 When to Change the Database Sizes from Their Defaults

The default values of the routing database sizes, together with the memory requirements for X.25 functions and NetWare IPX/AppleTalk routing, are given in Table 9–6.

Digital recommends that you do not change the routing database sizes except in the following circumstances:

- You know that the default values of one or more databases are not large enough. For example, you have configured the DECNIS to have 250 OSI reachable addresses, and the default value of the REACHABLE DESTINATIONS characteristic is 200.
- The DECNIS logs an event indicating that insufficient memory was available for the operation you require. For example, if the DECNIS logs an Adjacency Rejected event with a reason code of Maximum Router Adjacencies Exceeded when you try to enable a routing circuit, this could indicate that the value of the MAXIMUM ROUTER ADJACENCIES characteristic is too low.
- The memory occupied by the routing databases leaves insufficient memory for your X.25 functions, or AppleTalk/NetWare IPX routing.

9.31.6.1 Allow for Later Expansion

Note that you should ensure that the sizes of the routing databases are sufficient not only for the configuration in the master NCL script file, but also for any relevant NCL commands in the user NCL script files or any NCL commands that may be issued to the DECNIS when it is running.

For example, the value of the MAXIMUM REACHABLE DESTINATIONS characteristic might be sufficient at boot time, but you may not be able to subsequently add further OSI reachable addresses while the DECNIS is running.

9.31.7 Using the DECNIS Configurator to Change Routing Database Sizes

You cannot change the size of the routing databases on a running system. You must update the permanent configuration and then reboot the DECNIS.

Digital strongly recommends that you use the DECNIS configurator (text-based or Windows) to change the size of the routing databases. This is because the configurators can perform checks that help to prevent a configuration that requires more memory than is available. In addition, the configurators check, where possible, that the size of individual routing databases are sufficient for your configuration.

If you change the sizes of one or more routing databases without using one of the DECNIS configurators (for example, by placing NCL commands in the user NCL script files), you may exceed the amount of memory available on the DECNIS. If this happens, the DECNIS may remain unreachable after booting.

9.31.7.1 Using the DECNIS Text-based Configurator

The DECNIS text-based configurator asks you if you wish to adjust the memory allocated to the various routing databases.

- If you answer No ...

  The configurator chooses appropriate values for your configuration. In most cases, these will be the default values listed in Table 9–6. In some cases (for example, MAXIMUM REACHABLE DESTINATIONS), the configurator will use values according to information you provided earlier.

- If you answer Yes ...

  You can change the values of these characteristics (the default values are shown on the screen). The configurator will then check that the values you have chosen fit within the memory available on the DECNIS.

9.31.7.2 Using the clearVISN DECNIS Configurator

To change database sizes using the clearVISN DECNIS configurator, follow the steps below.

To size these databases ...	Follow these steps:
IP	1. On the Main Navigation window, click **IP Routing** 2. On the General tab page, click the **Database Sizing** button. The database sizing window appears.
DECnet/OSI	1. On the Main Navigation window, click **DNA IV/V + OSI** 2. Select the Database Sizing tab page.

9.31.8 Choosing New Values for the Routing Databases

9.31.8.1 Total Memory Available

The total memory available for your configuration depends on the version of management processor (MPC) installed on your DECNIS, as shown in the following table:

MPC	Memory Available
MPC–I	3500 Kbytes
MPC–II	11500 Kbytes
MPC–III	11500 Kbytes (16 MB version) 27500 Kbytes (32 MB version)

The memory allocated to the various routing databases, X.25 functions, and NetWare IPX and AppleTalk routing must not exceed this value.

You can use Table 9–6 to check if the DECNIS has sufficient unused memory to allow you to increase the size of a routing database without decreasing the size of others.

9.31.8.2 Choosing Which Database to Decrease

If there is insufficient DECNIS memory remaining to allow you to increase the size of a routing database, you can either reduce the X.25 functions, remove AppleTalk or NetWare IPX routing, or reduce the size of one or more routing databases by an appropriate amount.

To see which of the routing databases to reduce, follow these steps:

1. Issue the following command:

   ```
   NCL> SHOW ROUTING ALL ATTRIBUTES
   ```

   This will return the values of the status and characteristic attributes listed in Table 9–4.

2. For each characteristic attribute, examine the values of the corresponding status attributes, to identify characteristics that could be reduced without impacting operation of the DECNIS.

3. Use the DECNIS configurator (text-based or Windows) to make the adjustments required, and reboot the DECNIS.

4. You may need to repeat this process to fine-tune the sizes of the routing databases.

9.31.8.3 Example

The characteristic MAXIMUM ROUTER ADJACENCIES controls the number of Level 1 and Level 2 adjacencies the DECNIS can store. The number of adjacencies actually being stored is shown by the values of the following status attributes:

CURRENT L1 ROUTER ADJACENCIES
CURRENT L2 ROUTER ADJACENCIES

The peak values of the Level 1 and Level 2 router adjacencies since the DECNIS was last rebooted are shown by the values of the following status attributes:

PEAK L1 ROUTER ADJACENCIES
PEAK L2 ROUTER ADJACENCIES

If the value of the PEAK L1 ROUTER ADJACENCIES plus the value of the PEAK L2 ROUTER ADJACENCIES is much less than the value of the MAXIMUM ROUTER ADJACENCIES characteristic, it may mean that this characteristic could safely be reduced, so making more memory available for other purposes.

To test this, you could set the value of MAXIMUM ROUTER ADJACENCIES to a lower level, reboot the DECNIS, and monitor the values of the corresponding status attributes.

9.31.9 Memory Allocation on the DECNIS

Table 9–6 shows the amount of memory, in bytes, required for each configuration parameter listed, for X.25 functions, and for AppleTalk and NetWare IPX routing.

For example, each reachable destination requires 340 bytes. If you set the MAXIMUM REACHABLE DESTINATIONS characteristic to 50, then 50 x 340 = 17,000 bytes of the DECNIS routing memory are allocated for reachable destinations, and are not usable for any other purpose.

Network Connectivity

Table 9–6 assumes a network connectivity of 2 (that is, the values of the connectivity characteristics are 20). If you set the connectivity characteristics to a higher value, more memory will be required for each configuration parameter listed.

Table 9–6 DECNIS Memory Required for Each Configuration Parameter

Configuration Parameter	Minimum Value	Default	DECNIS Bytes Required for Each Unit Increment
Total memory available: 3500 Kbytes for MPC–I; 11500 Kbytes for MPC–II; 11500 Kbytes or 27500 Kbytes for MPC–III (depending on version)			
Router adjacencies	10	170	140
Endsystem adjacencies	10	5120	50
Manual adjacencies	0	60	230
Reachable destinations	0	200	340
DA adjacencies	0	160	280
L1 routing destinations	100	5280	80
L1 routers	10	100	80
L2 routing destinations	100	456	50
L2 routers	10	512	70
L1 LSPs[1]	10	400[1]	120
L2 LSPs[1]	10	1177[1]	120
L1 connectivity factor	10	20	700[2]
L2 connectivity factor	10	20	9800[2]
IP local adjacencies	1	50	100
IP reachable destinations	0	200	240
IP L1 destinations	100	250	100
IP L2 destinations	100	890	100
IP external destinations	100	500	100
IP L1 connectivity factor	10	20	–
IP L2 connectivity factor	10	20	–
OSPF Maximum Connected Areas	1	2	19,264

[1] The default LSP value depends on the other values in this table. The value shown in the **Default** column is the value that the DECNIS will calculate if the other values in this table are the defaults. If any of the other values are different, the DECNIS will calculate a different LSP default.

[2] This is the number of extra bytes required on changing from a value of 20 to a value of 25 (approximately 2.0 to 2.5 connections per router). This will be the value if all the other values in this table are at their default values.

(continued on next page)

Table 9–6 (Cont.) DECNIS Memory Required for Each Configuration Parameter

Configuration Parameter	Minimum Value	Default	DECNIS Bytes Required for Each Unit Increment
OSPF Average Connected Routers	1	10	680
OSPF Maximum Area Interfaces	1	3	948
OSPF Average Area Networks	1	40	480
OSPF Maximum Network Routers	1	5	320
OSPF Maximum System Networks	1	25	432
OSPF Maximum Boundary Routers	0	4	584
OSPF Maximum External Routes	0	50	224
OSPF Average External Connectivity	0	2	192
Maximum Destinations	1	300	132
Maximum Adjacencies	1	25	24
X.25, NetWare IPX, and AppleTalk Resources			
X.25 DTE/DCE	0	0	56,000
X.25 static routing circuit	0	0	23,000
X.25 dynamically assigned routing circuit	0	0	17,000
X.25 gateway virtual circuit	0	0	6,000
NetWare IPX networks[3]	0	0	140 + 50 per router per network
NetWare IPX services[3]	0	0	100 + 40 per service per server
Circuits with NetWare IPX[4]	0	0	280

[3] You can have up to 250 NetWare IPX circuits and 250 NetWare IPX services.

[4] You can have up to 32 tunnel circuits for NetWare IPX, and 32 tunnel circuits for AppleTalk.

(continued on next page)

Table 9–6 (Cont.) DECNIS Memory Required for Each Configuration Parameter

Configuration Parameter	Minimum Value	Default	DECNIS Bytes Required for Each Unit Increment
AppleTalk networks[5]	0	0	50
AppleTalk zones[5] [6]	0	0	30
AppleTalk nodes	0	0	40
Circuits with AppleTalk[4]	0	0	200

[4] You can have up to 32 tunnel circuits for NetWare IPX, and 32 tunnel circuits for AppleTalk.

[5] You can have up to 300 AppleTalk networks and 300 AppleTalk zones.

[6] AppleTalk zones that appear on multiple AppleTalk networks are counted separately.

10

IP Routing

10.1 Introduction

This chapter describes how to manage IP routing on the DECNIS. It describes the following protocols:

- Integrated IS–IS
- OSPF
- RIP
- BGP
- EGP

In addition, it describes the use of SNMP to monitor the IP information on the DECNIS.

10.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

Note

For any tasks that require you to disable a routing circuit, ensure that you do not disable the routing circuit that you are using to manage the DECNIS, unless there is an alternative route between the host on which you issue the NCL commands and the DECNIS.

10.2 Using the DECNIS as an IP Router

10.2.1 Routing Circuits

You can configure the DECNIS as an IP router. An IP router (sometimes called an IP gateway) is a system that routes IP packets within an IP network.

To configure the DECNIS as an IP router, you must set up routing circuits to connect with the IP network. These circuits can be of the following types:

- CSMA/CD (see Section 9.3)
- FDDI (see Section 9.4)
- HDLC (see Section 9.5)
- DDCMP (see Section 9.5)
- ATM Permanent (see Section 9.6)
- PPP (see Section 9.9)
- CHDLC (see Section 9.11)
- SMDS (see Section 9.12)
- VCP (see Section 8.9)
- X25 DA (see Section 9.17)
- X25 STATIC OUTGOING (see Section 9.18)
- X25 STATIC INCOMING (see Section 9.19)
- X25 PERMANENT (see Section 9.20)

The same circuit can be used for both IP routing and DECnet/OSI routing.

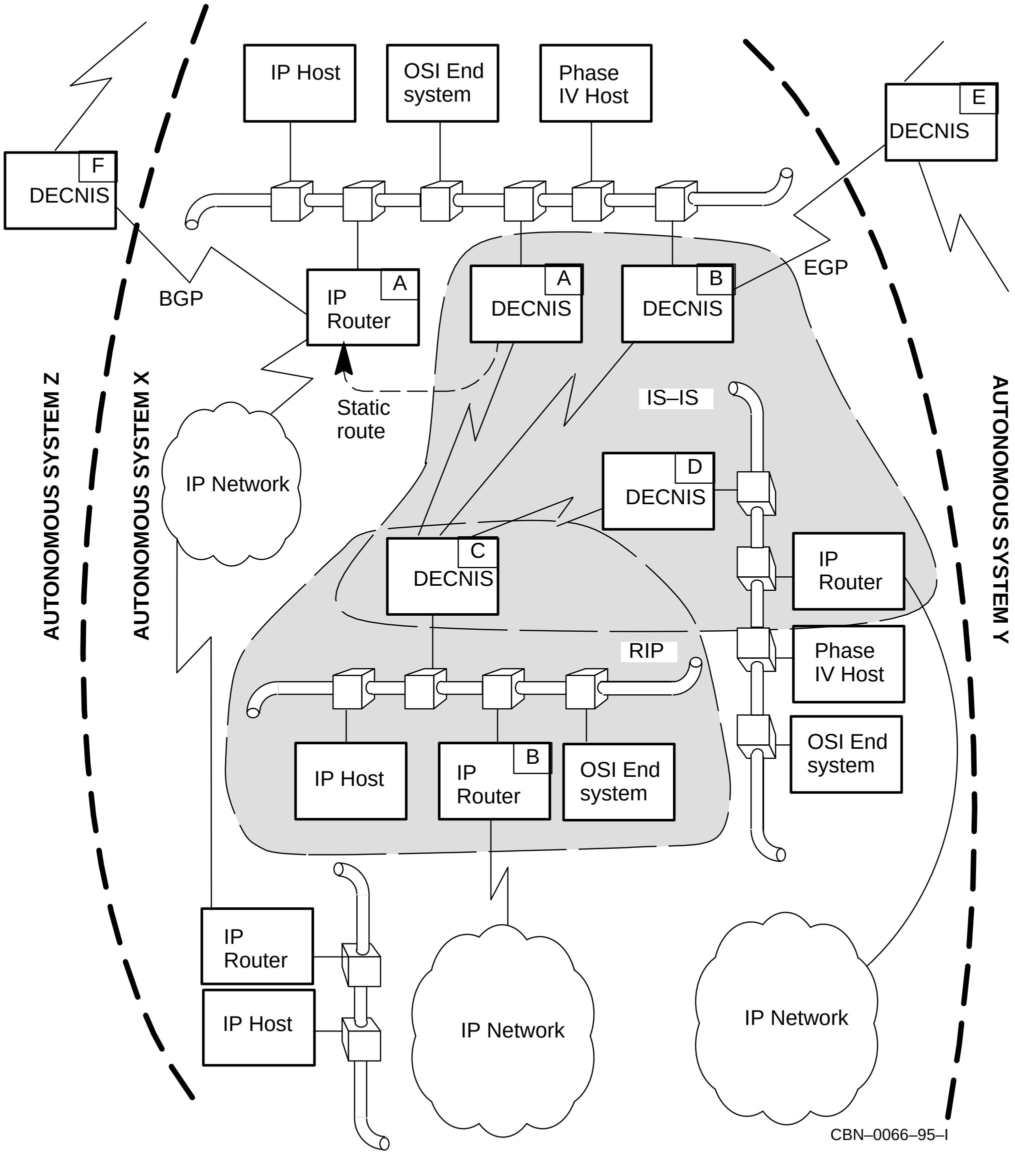
10.2.2 Restrictions on Using IP on HDLC and DDCMP Circuits

You cannot run IP routing on an HDLC or DDCMP routing circuit if you have set the attribute ROUTING CIRCUIT DNA NEIGHBOR to FALSE. This does not apply to any other type of routing circuit. For more information about the attribute DNA NEIGHBOR, refer to Section 16.16.3.

10.2.3 Example IP Network

Figure 10–1 shows an example IP network using DECNIS systems as IP routers.

Figure 10–1 Example IP Configuration



10.2.4 Forwarding IP Packets to Non-Digital Systems

Note that:

- On LANs and X.25 dynamically assigned routing circuits, the DECNIS can forward IP packets to IP hosts or IP routers from other vendors.
- On ATM Permanent, PPP, and CHDLC routing circuits the DECNIS can forward IP packets to IP hosts or IP routers from other vendors. However, the vendor system may require an IP address and subnet mask defined for the circuit. Refer to the vendor's documentation for more information.
- On HDLC, DDCMP, and X.25 static routing circuits, the DECNIS can only forward IP packets to other Digital IP routers.

10.2.5 Selecting an IP Protocol

Digital strongly recommends that you use only one IP routing protocol between any two routers. Wherever possible, use the Integrated IS–IS protocol (referred to in this manual as IS–IS), or the Open Shortest Path First (OSPF) protocol. Both meet the routing needs of large mesh networks that demand power, flexibility and reliability. Whichever routing protocol is chosen will depend on, for example, whether there is already an IP or DECnet/OSI network in place.

10.3 IP Protocols

10.3.1 Protocols Supported by DECNIS

The DECNIS supports the following IP routing protocols:

- Interior Gateway Protocols (IGPs):
  - Integrated IS–IS (defined in RFC 1195)
  - OSPF (defined in RFC 1247)
  - RIP (defined in RFC 1058)
- Exterior Gateway Protocols:
  - BGP (Version 4, defined in RFC 1771)
  - EGP (defined in RFCs 827, 888, and 904)

10.3.2 Autonomous Systems

A group of IP networks and routers controlled by a single administrative authority is called an autonomous system. This is analogous to a Routing Domain in DECnet/OSI routing.

Generally, the routers within an autonomous system communicate using an Interior Gateway Protocol. An Exterior Gateway Protocol is used to link autonomous systems.

Each autonomous system has an autonomous system number. This is a decimal number between 0 and 65,535, which is assigned to the autonomous system by the central authority for assigning IP network addresses.

10.4 IP Addressing

10.4.1 IP Address Format

IP addresses are 32-bit binary numbers. They are usually written as four decimal integers (one for each octet) separated by points. For example, the IP address 10001011 00111011 10101100 11100010 would be written as 139.59.172.226.

10.4.2 Subnet Masks

Each IP address normally has an associated subnet mask.

A subnet mask identifies the network address and subnetwork address parts of the IP address.

Format of Subnet Masks

A subnet mask is also a 32-bit binary number. It consists of contiguous 1s followed by contiguous 0s. The contiguous 1s indicate the address and subnet address fields of the IP address, and the contiguous 0s indicate the host field of the IP address.

The contiguous 1s in a subnet mask do not have to form a whole number of octets; for example, 255.224.0.0 is a valid subnet mask.

10.4.3 Example

Consider an IP address of 24.45.21.8.

As the first octet is in the range 1–126, this is a Class A network. Therefore, the first octet indicates the "pure" network part of the address.

Without a subnet mask, the last three octets would be taken to be the host address; that is, the host address would be 45.21.8.

Now, give this address a subnet mask of 255.255.0.0, that is, all binary 1s in the first two octets. This mask indicates that the second octet is a subnet address, rather than part of the host addres.

With the mask, this IP address is part of an IP subnetwork in which all IP addresses are of the form 24.45.*x.y*, where *x.y* identifies the host field of the IP address, and is different for each IP address.

With the mask, the host field is 21.8.

10.5 Assigning IP Addresses and Subnet Masks

10.5.1 Subnet Masks when Running RIP

If an IP network to which the DECNIS is connected is running RIP, you must use the same subnet mask for all IP interfaces that connect to this network.

If the DECNIS is connected to more than one IP network, only the interfaces to the same IP network must have the same subnet mask.

10.5.2 Subnet Masks when Running OSPF

If you are running OSPF, rather than RIP, in an IP network, you can implement variable length IP subnet masking. This provides flexibility in interpretation of the IP address bits, so that network growth and variations in configuration can be accommodated.

10.5.3 Multiple IP Addresses on a LAN Interface

If the IP interface is a LAN interface, you can assign several IP addresses to it: see Section 10.24.

10.5.4 LAN Routing Circuits

You must assign a unique IP subnet address to each DECNIS LAN routing circuit over which you want to run IP.

10.5.5 Point-to-Point Routing Circuits

If you run RIP or EGP over a point-to-point circuit, you must supply either an IP subnet address for the circuit, or a Neighbor IP address. Section 10.10.2 describes how to do this. For any other protocol type, it is not essential to supply an IP subnet address for a point-to-point circuit.

If you do not assign an IP subnet address to a point-to-point routing circuit, the DECNIS uses the Manual IP address. Section 10.6 describes how the Manual IP address is set.

Note that special rules apply to PPP routing circuits; see Section 10.5.6.

10.5.6 PPP Routing Circuits

The DECNIS supports the IP-Addresses option (3) of PPP, as described in RFC 1332 "The PPP Internet Control Protocol (IPCP)", May 1992.

10.5.6.1 Running PPP Between Two DECNIS Systems

If you are running a PPP circuit between two DECNIS systems, the PPP protocol automatically advertises the address of one DECNIS to the other DECNIS.

The address advertised by the PPP protocol is one of the following:

- If the PPP routing circuit has an IP subnet address, then the local address of that circuit is advertised. For example, if the IP subnet address is address=16.36.16.1, mask=255.255.255.0, the advertised address is 16.36.16.1.
- If the PPP routing circuit does not have an IP subnet address, then the SYSTEM IP ADDRESS status attribute of Routing is advertised.

System IP Address

The System IP Address status attribute is derived from the Manual IP Address characteristic. Section 10.6.2 describes how to set the Manual IP Address.

For example, if the DECNIS has two LAN circuits with subnet addresses of address=16.36.16.2, mask=255.255.255.0 and address=16.36.17.6, mask=255.255.255.0. and you wish PPP to advertise the address of the DECNIS as 16.36.17.6, you should set the Manual IP Address to 16.36.17.6.

Finding the Negotiated Address

To find the address advertised by PPP on the remote DECNIS, enter the following command:

```
NCL>  SHOW NODE remote-system ROUTING CIRCUIT circuit-name-
_NCL>  NEGOTIATED NEIGHBOR IP ADDRESS
```

10.5.6.2 Running PPP between a DECNIS and a Non-Digital Router

If you have a PPP link between a DECNIS and a non-Digital router, the other router may not support the IP-Addresses option (3). This means that the PPP protocol will not automatically advertise the addresses of the systems at either end of the link.

In this situation, Digital recommends that you set up a Neighbor IP Address. This is the address of the system at the other end of the circuit. To set a Neighbor IP Address, enter the following commands:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name
NCL> SET ROUTING CIRCUIT circuit-name NEIGHBOR IP ADDRESS=a.b.c.d
NCL> ENABLE ROUTING CIRCUIT circuit-name
```

10.5.7 Procedure for Assigning IP Subnet Addresses

To assign IP addresses to an IP routing circuit (interface), follow these steps:

1. Disable the routing circuit corresponding to the IP interface:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit-name
   ```

2. Set the IP address and subnet mask for the interface:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> SUBNET ADDRESS {ADDRESS=a.b.c.d, MASK=e.f.g.h}
   ```

   where *a.b.c.d* is the IP address of the interface, and *e.f.g.h* is its subnet mask.

3. Enable the routing circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

10.6 Manual IP Address

This section explains the function of the manual IP address (also known as a system IP address), and describes how to set it up.

10.6.1 Source Address in IP Packets

All IP packets transmitted by the DECNIS must contain a source IP address. This is normally the IP address of the routing circuit over which the packet is transmitted.

If you have not assigned an IP address to the circuit from which the packet is transmitted, the source address contained in the packet will be the MANUAL IP ADDRESS for the DECNIS. The manual IP address is a default IP address for the DECNIS.

You must set up a manual IP address.

10.6.2 Setting a Manual IP Address

Set a manual IP address as follows:

```
NCL> SET ROUTING MANUAL IP ADDRESS a.b.c.d
```

where *a.b.c.d* is one of the IP addresses used on one of the DECNIS system's routing circuits.

You can also supply the manual IP address using the DECNIS configurators:

- On Windows 95/NT load hosts, in the clearVISN DECNIS configurator, under **IP Routing**.
- On other load hosts:
  - If you are using BOOTP loading, in the load-host configurator.
  - If you are using MOP loading, in the Routing section of the DECNIS text-based configurator.

10.7 Using the Integrated IS–IS Protocol for IP Routing

10.7.1 What Is IS–IS?

The Integrated IS–IS protocol allows both OSI and IP routing information to be propagated in the same link state packet.

The DECNIS will run the Integrated IS–IS protocol on any routing circuit that uses the link state routing algorithm. However, you can choose not to include IP routing information in such packets, by specifying in the DECNIS configurator (text-based or Windows) that you do not wish to use IS–IS for IP routing.

10.7.2 Routing Algorithm

Check that the routing circuits that are to use the IS–IS protocol are running the link state routing algorithm as follows:

- If the DECNIS is a Level 1 router (see Section 9.24), enter the following command:

  ```
  NCL> SHOW ROUTING MANUAL L1 ALGORITHM
  ```

  If the value returned by this command is LINK STATE, you can use the IS–IS protocol on any routing circuit.

  If the value returned by this command is ROUTING VECTOR, you cannot use the IS–IS protocol on any routing circuit.

- If the DECNIS is a Level 2 router (see Section 9.24), enter the following command:

  ```
  NCL> SHOW ROUTING MANUAL L2 ALGORITHM
  ```

  If the value returned by this command is LINK STATE, you can use the IS–IS protocol on any routing circuit.

If the value returned by this command is ROUTING VECTOR, you cannot use the IS–IS protocol on any routing circuit, unless the circuit will connect to another router in the same area as the DECNIS, and the DECNIS is running the link state algorithm at Level 1.

10.7.3 Procedure

You can specify whether or not you want to use the IS–IS protocol for IP routing in the DECNIS configurator (Windows or text-based).

For any LAN circuits on which you wish to run IS–IS, you must assign IP addresses and subnet masks. You do not need to assign IP addresses/subnet masks, nor neighbor IP addresses, for point-to-point circuits running IS–IS.

10.8 Using the OSPF Protocol

This section describes the main features of the OSPF protocol.

10.8.1 Introduction to OSPF

Open Shortest Path First (OSPF) is an Interior Gateway Protocol (IGP). It is an internet routing protocol which uses link-state or distributed database technology. This means that the internal routing algorithm can select the best (least costly) path for routing, and routing attributes can be dynamically changed by network management to accommodate fluctuations in traffic flow and expansion of the internetworks. The protocol responds rapidly to network topology changes, yet involves only small amounts of routing protocol traffic, allowing business traffic to flow freely and rapidly.

10.8.2 OSPF Organization

Autonomous Systems

All OSPF routers are grouped within an autonomous system. This means that a group of OSPF routers can exchange routing information via a common routing protocol. IP packets are routed and forwarded based on the IP destination address. The protocol detects topological changes, failures and additions within the autonomous system, calculates the best alternative route, and transmits packets accordingly.

OSPF Router Databases

Each router maintains its own topological database and knows the state of the autonomous system and its reachable neighbors, constantly updating the autonomous system with its current state.

OSPF Areas

Within the autonomous system, sets of networks can be grouped together in areas. Each area's topology is hidden from the rest of the autonomous system, thus reducing routing traffic.

OSPF Backbone

Every OSPF routing domain autonomous system that contains more than one area *must* have a backbone. This is a special area with area ID 0.0.0.0, and must be contiguous. Where the backbone is no longer contiguous, *virtual links* must be established to restore backbone continuity.

OSPF Routing Algorithm

All routers have the same routing algorithm and construct a map of shortest paths to each destination in the autonomous system. When there are several equal-cost routes to a destination, traffic is distributed equally among them.

Intra-area Routing

Routing is called *intra-area* routing when the source and destination reside in the same area.

Interarea Routing

Routing is called *interarea* routing when the source and destination reside in different areas.

External Routing

Externally derived routing information from other routing protocols can be filtered into an OSPF autonomous system.

Similarly, OSPF routers can propagate route information outside an autonomous system.

10.8.3 Authentication

All OSPF protocol exchanges may be authenticated on an area by area basis. This means that only trusted routers can participate in the autonomous system's routing. With the current version of OSPF, each area may be set up with or without protocol exchange authentication.

10.8.4 OSPF Interface Types

An OSPF interface is the connection between an OSPF router and a network, or between two OSPF routers.

Note

The term ‘Interface’ used here is synonymous with the Digital term ‘logical circuit’.

There are four OSPF interface types:

- Broadcast

  Networks supporting more than two attached routers, together with the capability to address a single physical message to all of the attached routers. Neighboring routers are discovered dynamically on these networks using OSPF’s Hello Protocol. The Hello Protocol itself takes advantage of the broadcast capability. The protocol makes further use of multicast capabilities, if they exist. An Ethernet is an example of a broadcast network.

- Nonbroadcast

  Networks supporting more than two attached routers, but having no broadcast capability. As with broadcast networks, neighboring routers are discovered using the Hello Protocol. However, due to the lack of broadcast capability, some configuration information is necessary for the correct operation of the Hello Protocol. On these networks, OSPF protocol packets that are normally multicast need to be sent to each neighboring router, in turn. An X.25 Packet Switched Data Network (PSDN) is an example of a nonbroadcast network.

- Point-to-point

  A network that joins a single pair of routers. For example, a PPP link.

- Virtual link

  An independent logical path which must be created when the) backbone is no longer contiguous, to restore backbone continuity.

10.8.5 Router Classification

Within an autonomous system, routers can be split into OSPF areas. Figure 10–2 shows the different functions of an OSPF router. There are four types of router, as described in RFC 1247:

- Internal Router

  A router with all directly connected networks belonging to the same area. Routers with only backbone interfaces also belong to this category. These routers run a single copy of the basic routing algorithm.

- Area Border Router

  A router that attaches to multiple areas. Area Border Routers run multiple copies of the basic routing algorithm, one copy for each attached area and an additional copy for the backbone. The backbone in turn distributes the information to other areas.

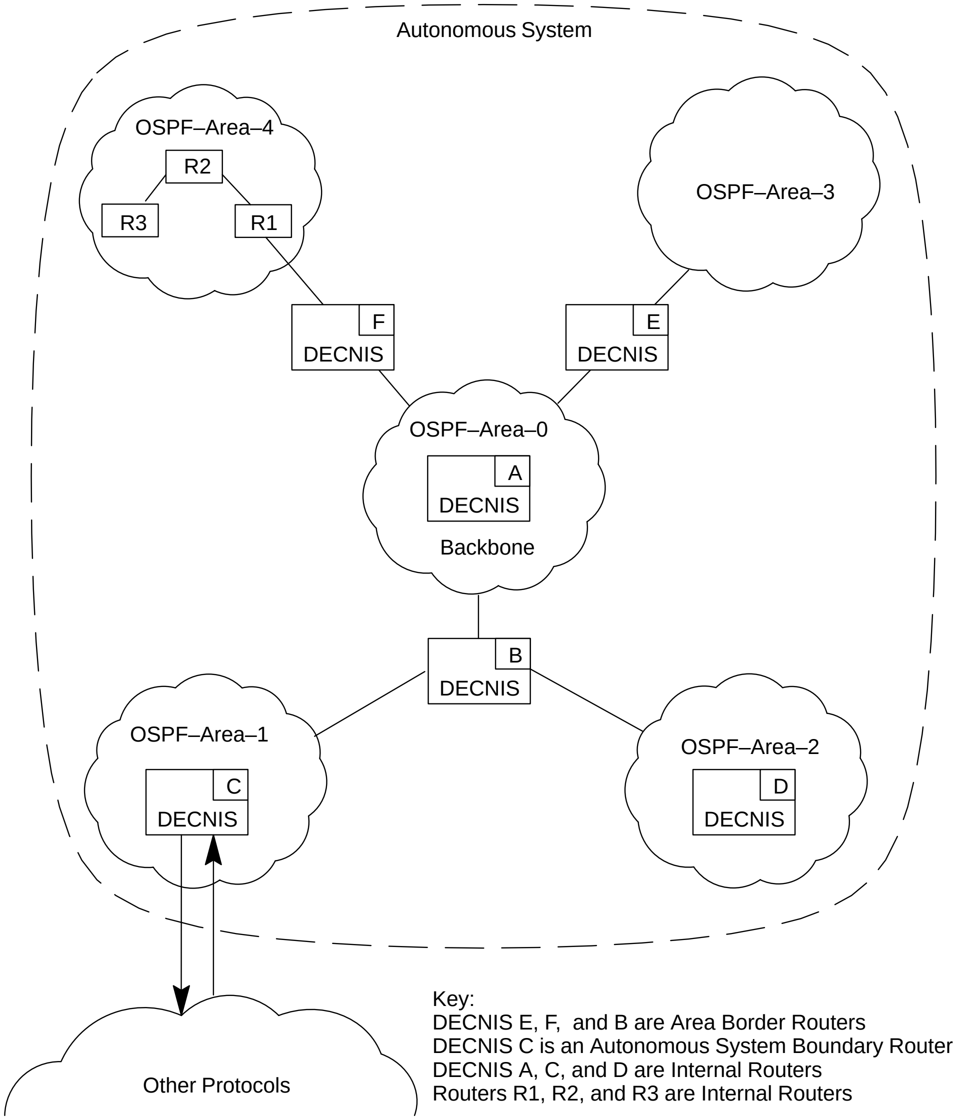
- Backbone Router

  A router that has an interface to the backbone. This includes all routers that interface to more than one area (that is, Area Border Routers). However, Backbone Routers do not have to be Area Border Routers. Routers with all interfaces connected to the backbone are considered to be Internal Routers.

- Autonomous System Boundary Router

  A router that can advertise routing information from other routing protocols throughout the OSPF Autonomous System. The path to each Autonomous System Boundary Router is known by every router in the autonomous system. Autonomous System Boundary Routers may be Internal Routers or Area Border Routers, and may or may not participate in the backbone.

Figure 10–2 Different Functions of an OSPF Router



Key:
DECNIS E, F, and B are Area Border Routers
DECNIS C is an Autonomous System Boundary Router
DECNIS A, C, and D are Internal Routers
Routers R1, R2, and R3 are Internal Routers

CBN–0054–93–I

10.9 Configuring the DECNIS to Use the OSPF Protocol

This section describes how to configure the DECNIS to use the OSPF protocol.

10.9.1 ROUTING CONTROL PROTOCOL Entities for OSPF

On the DECNIS, OSPF is managed by ROUTING CONTROL PROTOCOL entities. There are two types of ROUTING CONTROL PROTOCOL entity for OSPF:

- Type OSPF GENERAL

  One ROUTING CONTROL PROTOCOL of Type OSPF GENERAL is needed per router. This defines the OSPF instance and contains the database sizing parameters. You can use the DECNIS configurator (Windows or text-based) to set it up.

- Type OSPF AREA

  One ROUTING CONTROL PROTOCOL of type OSPF AREA is needed per OSPF area. This defines an OSPF area, and is the parent to which the interfaces (logical circuits) are attached. You can use the clearVISN DECNIS configurator to set up CONTROL PROTOCOL entities of type OSPF AREA. You cannot use the DECNIS text-based configurator to set up entities of this type.

10.9.2 Using the DECNIS Text-Based Configurator to Set Up OSPF

The DECNIS text-based configurator does not fully support OSPF. However, if you request OSPF, the configurator will set up a ROUTING CONTROL PROTOCOL of type OSPF GENERAL.

Also, if you request OSPF, you are automatically asked if you want the DECNIS to become an Autonomous System Boundary Router.

10.9.2.1 Procedure

In the Routing Section, follow these steps:

1. Select OSPF as an IP protocol.

   The following line is added to the master NCL script file:

   ```
   CREATE ROUTING CONTROL PROTOCOL OSPF-General TYPE OSPF GENERAL
   ```

2. You are now asked if you want the DECNIS to be an Autonomous System Boundary Router. If you select Yes, the following line is added to the master NCL script file:

   ```
   SET ROUTING CONTROL PROTOCOL OSPF-General OSPF AUTONOMOUS SYSTEM -
   BOUNDARY ROUTER TRUE
   ```

10.9.3 Using the clearVISN DECNIS Configurator to Set Up OSPF

The clearVISN DECNIS configurator provides support for OSPF that enables you to do the following:

- Set up ROUTING CONTROL PROTOCOLs of type OSPF AREA or OSPF GENERAL.
- Set up the DECNIS as an Autonomous System Boundary Router.
- Specify OSPF areas, the circuits that belong to them, and whether the area is a stub area.

10.9.3.1 Procedure

Follow these steps:

1. On the Main Navigation window, click IP Routing.
2. On the IP tab page, tick the OSPF box.
3. On the Circuits tab page, select a circuit that you want to use for OSPF, click the box **Run IP over this circuit** and supply an IP address.
4. Enter the required information on the OSPF tab page.
5. Enter the required information on the OSPF Circuits tab page.

Refer to the configurator online help for details.

10.9.4 Using NCL to Create ROUTING CONTROL PROTOCOL Entities of Type OSPF AREA

The following examples show how to set up ROUTING CONTROL PROTOCOL entities of type OSPF AREA for different network configurations.

- Example 1—Minimum OSPF configuration, one area with two interfaces.
- Example 2—Multiple OSPF areas, two areas plus backbone.
- Example 3—OSPF backbone connection using virtual links.
- Example 4—Password Authentication and setting interface password.

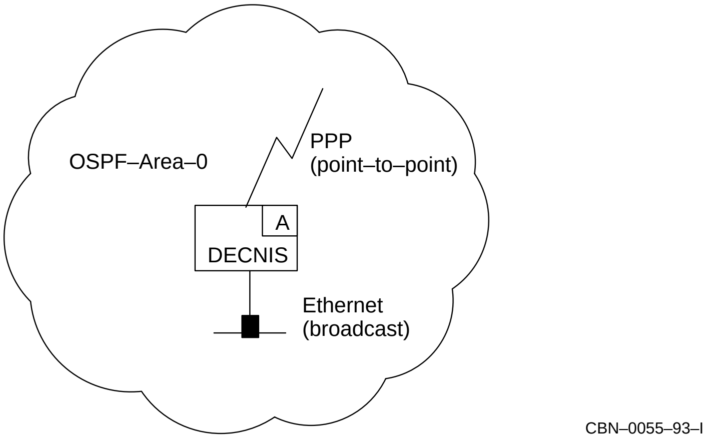
You can add any of the NCL commands in the examples to the user NCL script files. For details about the user NCL script files, refer to Section 1.5.6.

10.9.4.1 Example 1: Setting Up a Minimum OSPF Configuration

Figure 10–3 shows a minimum OSPF configuration. DECNIS A is an OSPF Internal Router in area 0.0.0.0 and has two OSPF interfaces:

- An Ethernet broadcast interface
- A PPP point-to-point interface

Figure 10–3 OSPF Minimum Configuration



Follow these steps:

1. Create the area CONTROL PROTOCOL entity, and call it OSPF-Area-0:

```
NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-0 TYPE OSPF AREA
```

2. Create the OSPF broadcast and point-to-point interfaces (logical circuits) and give them the same names as the associated routing circuits:

```
NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-0 -
_NCL> LOGICAL CIRCUIT l601-3-0 TYPE BROADCAST
```

```
NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-0 -
_NCL> LOGICAL CIRCUIT w622-5-0 TYPE POINT TO POINT
```

3. Associate the area with the OSPF General Entity:

```
NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-0 -
_NCL> OSPF GENERAL INSTANCE OSPF-General
```

4. Set up the area ID:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-0 -
   _NCL> OSPF AREA ID 0.0.0.0
   ```

5. Set the OSPF PRIORITY if you do not want the default priority of 1. Values 1—255 mean that a router is eligible to become the Designated Router. A value of 0 means that it cannot become the Designated Router.

   In this example, DECNIS A cannot become the Designated Router for the network attached to interface l601-3-0.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-0 -
   _NCL> LOGICAL CIRCUIT l601-3-0 OSPF PRIORITY 0
   ```

6. Configure the OSPF interfaces (logical circuits). Each logical circuit must be associated with a physical routing circuit:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-0 -
   _NCL> LOGICAL CIRCUIT l601-3-0 CIRCUIT ROUTING CIRCUIT l601-3-0
   ```

   ```
   NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-0 -
   _NCL> LOGICAL CIRCUIT w622-5-0 CIRCUIT ROUTING CIRCUIT w622-5-0
   ```

7. Enable the OSPF-General CONTROL PROTOCOL entity:

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-General
   ```

8. Enable the area and its circuits:

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-0
   ```

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-0 -
   _NCL> LOGICAL CIRCUIT l601-3-0
   ```

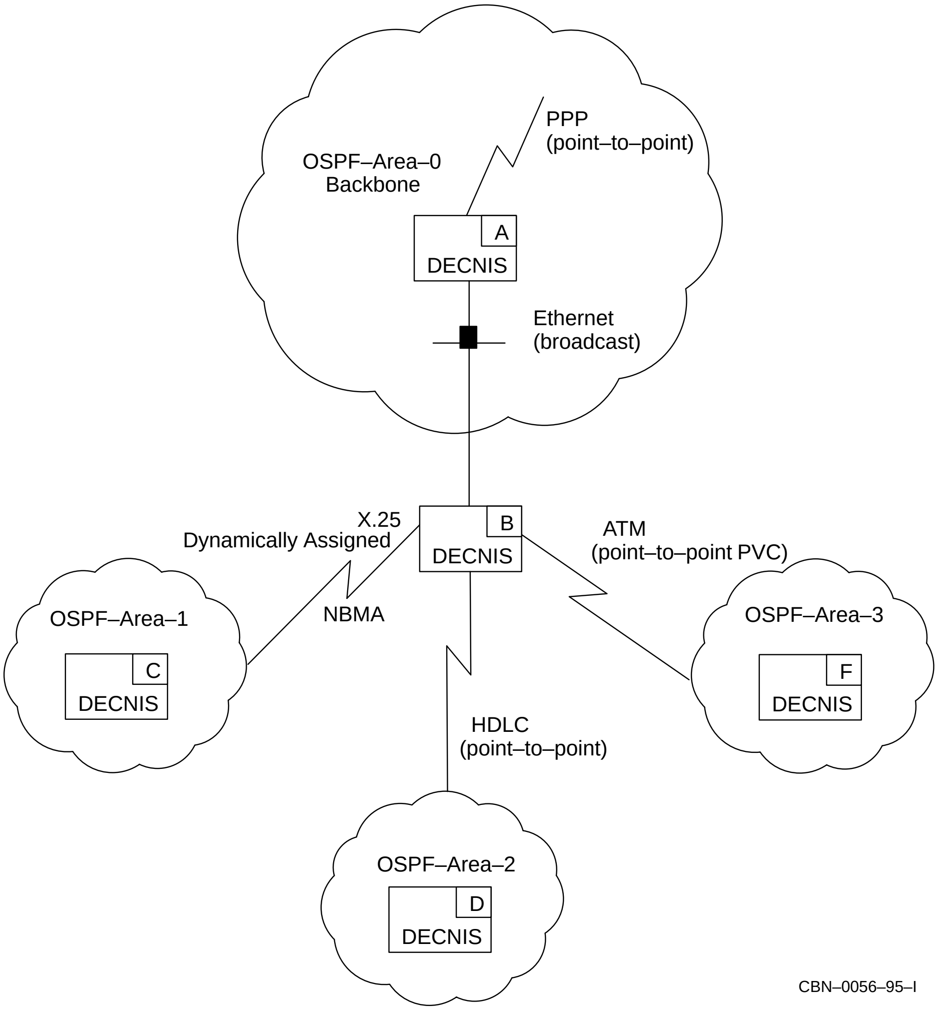
   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-0 -
   _NCL> LOGICAL CIRCUIT w622-5-0
   ```

10.9.4.2 Example 2: Setting Up an OSPF Multiple Areas Configuration

Figure 10–4 shows a configuration with multiple areas. DECNIS B is an OSPF Area Border Router. It attaches areas 1.1.1.1, 2.2.2.2, and 3.3.3.3 to backbone area 0.0.0.0, using four interfaces:

- An Ethernet broadcast interface to area 0.0.0.0
- An X.25 NBMA interface to area 1.1.1.1
- An HDLC point-to-point interface to area 2.2.2.2
- An ATM Permanent point-to-point interface to area 3.3.3.3

Figure 10–4 OSPF Multiple Area Configuration



Follow these steps:

1. Create all areas including the Backbone area:

```
NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-0 TYPE OSPF AREA

NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-1 TYPE OSPF AREA

NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-2 TYPE OSPF AREA

NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-3 TYPE OSPF AREA
```

2. Create the OSPF interfaces (logical circuits) and give them the same names as the associated routing circuits.

```
NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-0 -
_NCL> LOGICAL CIRCUIT l602-3-1 TYPE BROADCAST

NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-1 -
_NCL> LOGICAL CIRCUIT RX25-DA TYPE NBMA

NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-2 -
_NCL> LOGICAL CIRCUIT w618-4-0 TYPE POINT TO POINT

NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-3 -
_NCL> LOGICAL CIRCUIT w631-7-0 TYPE POINT TO POINT
```

3. DECNIS B is eligible to become the Designated Router on the attached NBMA network. It therefore requires a configured list of all its neighbors on the NBMA network. It takes the default OSPF PRIORITY of 1 (see Section 10.9.5 for further information about Designated Routers):

```
NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-1 -
_NCL> LOGICAL CIRCUIT RX25-DA -
_NCL> ADJACENCY decnis-C OSPF NEIGHBOR ADDRESS 16.36.16.100
```

4. Associate each area with the OSPF General Entity:

```
NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-0 -
_NCL> OSPF GENERAL INSTANCE OSPF-General

NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-1 -
_NCL> OSPF GENERAL INSTANCE OSPF-General

NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-2 -
_NCL> OSPF GENERAL INSTANCE OSPF-General

NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-3 -
_NCL> OSPF GENERAL INSTANCE OSPF-General
```

5. Set the area IDs for each area:

```
NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-0 -
_NCL> OSPF AREA ID 0.0.0.0

NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-1 -
_NCL> OSPF AREA ID 1.1.1.1

NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-2 -
_NCL> OSPF AREA ID 2.2.2.2
```

```
NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-3 -
_NCL> OSPF AREA ID 3.3.3.3
```

6. Configure the OSPF interfaces (logical circuits). Each logical circuit must be associated with a physical routing circuit:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-0 -
   _NCL> LOGICAL CIRCUIT 1602-3-1 CIRCUIT ROUTING CIRCUIT 1602-3-0

   NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-1 -
   _NCL> LOGICAL CIRCUIT RX25-DA CIRCUIT ROUTING CIRCUIT RX25-DA

   NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-2 -
   _NCL> LOGICAL CIRCUIT w618-4-0 CIRCUIT ROUTING CIRCUIT w618-4-0

   NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-3 -
   _NCL> LOGICAL CIRCUIT w631-7-0 CIRCUIT ROUTING CIRCUIT w631-7-0
   ```

7. Enable the OSPF-General CONTROL PROTOCOL entity:

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-General
   ```

8. Enable the OSPF-Area CONTROL PROTOCOL entities and their circuits:

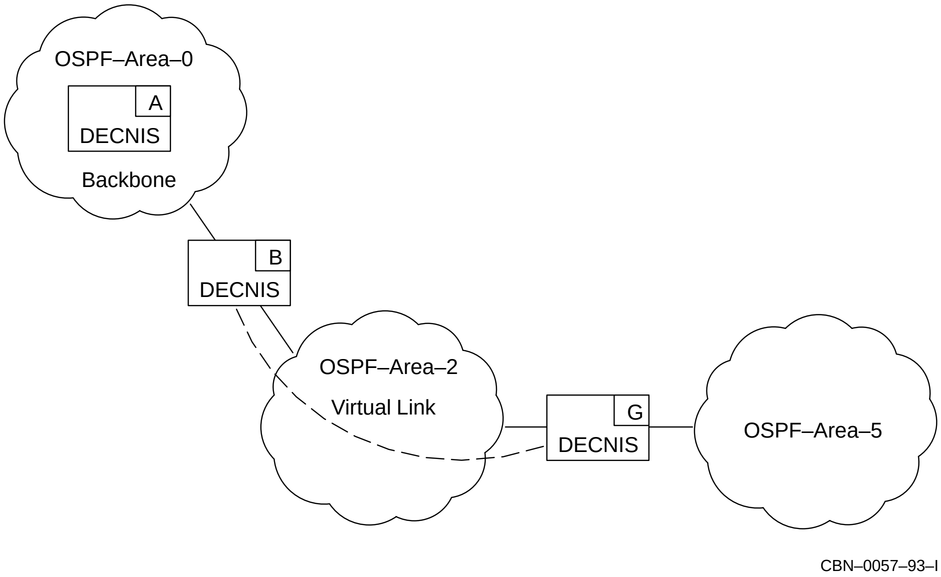
   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-0

   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-1

   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-2

   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-3

   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-0 -
   _NCL> LOGICAL CIRCUIT l602-3-1

   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-1 -
   _NCL> LOGICAL CIRCUIT RX25-DA

   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-2 -
   _NCL> LOGICAL CIRCUIT w618-4-0

   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-3 -
   _NCL> LOGICAL CIRCUIT w631-7-0
   ```

10.9.4.3 Example 3: Connecting the OSPF Backbone Using Virtual Links

Figure 10–5 shows a virtual link configuration. A virtual link is used to connect two disjointed sections of the backbone area.

DECNIS G is an Area Border Router of OSPF-Area-5. It is not directly connected to the rest of the backbone area. Instead, it uses a virtual link to DECNIS B, with OSPF-Area-2 as the Transit Area. DECNIS B is the Area Border Router between OSPF-Area-2 and OSPF-Area-0.

Figure 10–5 Using a Virtual Link to Connect to the OSPF Backbone



The virtual link must be configured on both Area Border Routers: DECNIS B and DECNIS G.

Use the following commands to configure DECNIS B:

1. Create the virtual circuit and call it ‘virtual-1’:

```
NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-0 -
_NCL> LOGICAL CIRCUIT virtual-1 TYPE VIRTUAL
```

2. Set the Router ID of the virtual neighbor (DECNIS G, Router ID: 16.36.16.180) and the Transit Area (OSPF-Area-2):

```
NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-0 LOGICAL CIRCUIT virtual-1 -
_NCL> OSPF VIRTUAL NEIGHBOR ROUTER ID 16.36.16.180, -
_NCL> OSPF TRANSIT AREA OSPF-Area-2
```

3. Enable the logical circuit:

```
NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-0 LOGICAL CIRCUIT virtual-1
```

4. Follow the same procedures to configure DECNIS G. The Virtual Neighbor Router ID is the Router ID of DECNIS B, which has an address of 17.36.16.207:

```
NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-Area-0 -
_NCL> LOGICAL CIRCUIT virtual-1 TYPE VIRTUAL

NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-0 LOGICAL CIRCUIT virtual-1 -
_NCL> OSPF VIRTUAL NEIGHBOR ROUTER ID 17.36.16.207, -
_NCL> OSPF TRANSIT AREA OSPF-Area-2

NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-Area-0 LOGICAL CIRCUIT virtual-1
```

10.9.4.4 Example 4: Procedure for Authentication

In this example, you are configuring an OSPF area to use simple password authentication, and then setting a password on an interface.

1. Set Area-0 to use simple password authentication. You can only do this whilst the area is disabled.

```
NCL> SET ROUTING CONTROL PROTOCOL OSPF-AREA-0 -
_NCL> OSPF AUTHENTICATION SIMPLE PASSWORD
```

2. Configure the l601-3-0 interface to use a password of "OSPFTest". You can have different passwords on each interface. Note that the password is an 8-byte hexadecimal number:

```
NCL> SET ROUTING CONTROL PROTOCOL OSPF-AREA-0 LOGICAL CIRCUIT l601-3-0 -
_NCL> OSPF AUTHENTICATION KEY %x4F53504674657374
```

10.9.5 Designated Routers

Designated Router Definition

In OSPF terms, a Designated Router is the router with the highest Router Priority. It originates a network links advertisement on behalf of the network, and is adjacent to all other routers on the network.

Backup Designated Router

The Backup Designated Router is also adjacent to all other routers on the network, and becomes the Designated Router should the previous Designated Router fail. At the time of (Backup) Designated Router election, this is the router with the second highest Router Priority.

Router Priority

The Router Priority is contained in a router’s Hello Packet. The Hello Protocol itself elects both the Designated Router and the Backup Designated Router based on the Router Priority. Router Priority values are 0–255. A Router Priority of 0 means that the router is not eligible to become the designated Router. The highest Router Priority is 255.

Designated Routers on Nonbroadcast Networks

On nonbroadcast networks such as NBMA, only eligible Designated Routers can become the Designated Router or Backup Designated Router.

A router becomes eligible if it has a configured list of all neighbors on the network, and each listed neighbor is labelled with its Designated Router eligibility.

On nonbroadcast networks, ensure that there are at least two eligible Designated Routers so that one is elected Designated Router, and one is elected Backup Designated Router.

10.9.6 Using NBMA on Ethernet

There are two advantages of using NBMA on an Ethernet:

- There is less routing traffic since routers only become neighbors with the Designated Router and the Backup Designated Router.
- There is a higher level of security. Because neighbors have to be in a configured list, you can decide which other systems you become adjacent to.

10.9.7 Alternative Subnetworks

You can specify more than one IP address on a LAN routing circuit. These are called alternative subnet addresses (see Section 10.24 for details).

Although OSPF itself only understands one IP address per interface, DECNIS OSPF will advertise these alternative subnet addresses as stub networks in its Router LSA packets.

10.9.8 Unused Interfaces

Unlike IS–IS, OSPF does not advertise the IP addresses of interfaces that are not configured for OSPF. By configuring the interface as an NBMA interface without any neighbors, the DECNIS *will* advertise the interface address. However, the DECNIS will not:

- Become adjacent to any OSPF routers on the interface
- Send any OSPF routing control packets
- Process any OSPF routing control packets it receives on that interface

10.10 Configuring the DECNIS to Use the RIP Protocol

This section describes how to set up the DECNIS to use the RIP protocol.

10.10.1 RIP Instances

You are allowed to create more than one instance of the RIP protocol on the DECNIS. This enables you to control the reachability between subnets running RIP.

Figure 10–6 gives an example of how RIP can be set up. There are two RIP instances set up on DECNIS B: RIP-1 on Subnet A and Subnet B and RIP-2 on Subnet C. By default, there is no propagation of routes learnt from one RIP instance into another. This means that routes learnt from RIP-1 subnets will not be advertised in the RIP-2 subnets.

RIP Propagation

By using different RIP instances and propagating between them, you have greater control over the RIP traffic on your network, for example:

- In Figure 10–6, if you later wanted to set up reachability between Subnet C and Subnets A and B, you could set up DECNIS B to propagate RIP-1 routes into the RIP-2 subnet.
- In Figure 10–6, you might want IP reachability from Subnet C to Subnet A, but not to Subnet B. In this situation, you could set up RIP-3 on Subnet A and set up DECNIS B to propagate RIP-3 routes to RIP-2.

For more details about setting up route propagation, refer to Section 11.6.

10.10.2 RIP Over Point-to-Point Circuits

To run RIP over a point-to-point circuit, other than a PPP circuit, you must set up circuit addressing in ONE of the following ways:

- Assign a subnet address to the point-to-point circuit by entering the following commands:

  ```
  NCL> DISABLE ROUTING CIRCUIT circuit-name
  ```

  ```
  NCL> SET ROUTING CIRCUIT circuit-name SUBNET ADDRESS {ADDRESS=a.b.c.d, -
  _NCL> MASK=e.f.g.h}
  ```

  ```
  NCL> ENABLE ROUTING CIRCUIT circuit-name
  ```

  The remote system must be in the same subnet.

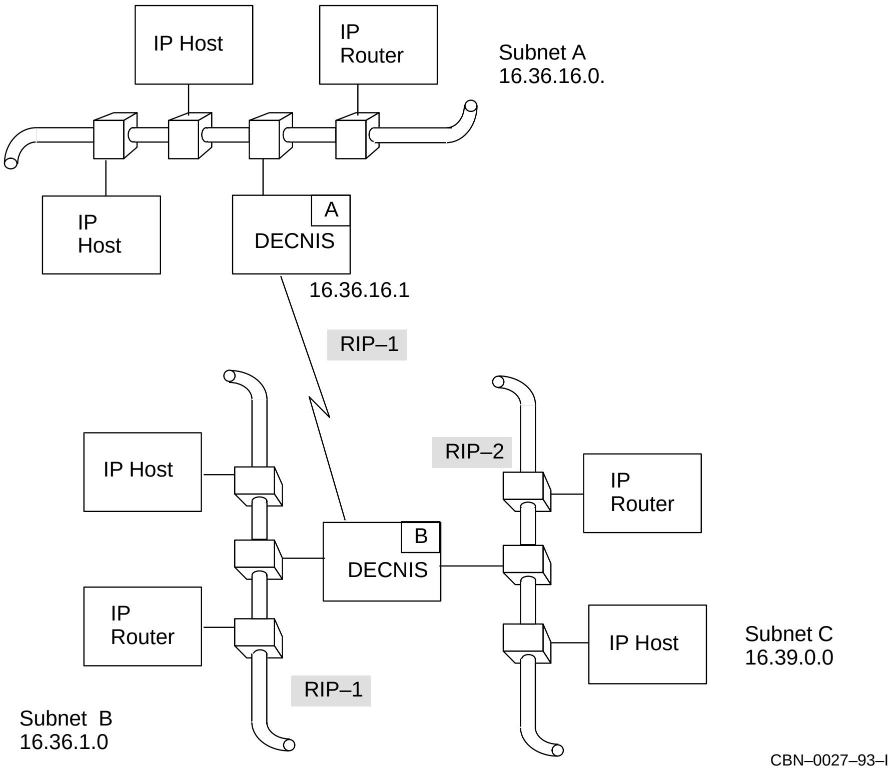
- Set a Neighbor IP Address. This is the address of the system at the other end of the circuit. Enter the following command:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name
NCL> SET ROUTING CIRCUIT circuit-name NEIGHBOR IP ADDRESS=a.b.c.d
NCL> ENABLE ROUTING CIRCUIT circuit-name
```

Note that you must not set up both a subnet address and a Neighbor IP Address on the same circuit.

If you are running RIP over a PPP circuit, refer to Section 10.5.6 for details about IP addressing.

Figure 10–6 Using RIP in an IP Network



10.10.3 Procedure for Setting Up RIP

Configure the DECNIS to use the RIP protocol as follows:

1. Set up a Subnet address or a Neighbor IP Address on any point-to-point routing circuits you want to use (see Section 10.10.2).

2. Create a ROUTING CONTROL PROTOCOL entity for each RIP instance that you want to create. Enter the following command:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-prot-name TYPE IP RIP
   ```

   where *control-prot-name* is a name for this RIP instance.

   For example:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL rip-1 TYPE IP RIP
   ```

3. Associate each RIP instance with one or more subnets:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> IP SUBNET {{ADDRESS=a.b.c.d, MASK=e.f.g.h},...}
   ```

   where:

   - A particular subnet can only appear on one RIP control protocol.
   - If the RIP instance is over a LAN circuit, *a.b.c.d* is the address of the subnet and *e.f.g.h* is the subnet mask.

     For example, in Figure 10–6, RIP-1 over Subnet B has the address and mask: 16.36.1.0, 255.255.255.0.
   - If the RIP instance is over a point-to-point circuit (apart from a PPP circuit), *a.b.c.d* is either the Subnet address or the Neighbor IP Address of the circuit and *e.f.g.h* is the mask. If you specify a Neighbor IP address, the mask should be 255.255.255.255.

     For example, in Figure 10–6, RIP-1 over the point-to-point circuit to Subnet A is associated with a Neighbor IP address. This means that *a.b.c.d* is 16.36.16.1 and *e.f.g.h* is 255.255.255.255.

     If the RIP instance is over a PPP circuit, refer to Section 10.5.6 to determine what address is required.

4. Specify whether RIP messages are to be sent, received or both.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name RIP STATE state
   ```

   where *state* is one of the following:

   SEND AND RECEIVE
   SEND ONLY
   RECEIVE ONLY

5. Specify whether Poisoned Reverse applies to the RIP instance.

   By default, the DECNIS operates RIP Poisoned Reverse. This means that RIP routes that are received by the DECNIS are advertised back to the subnet from which they were received marked "unreachable".

   If you do not want the RIP instance to use Poisoned Reverse, issue the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name POISONED REVERSE -
   _NCL> FALSE
   ```

6. Specify any IP sinks.

   By default, RIP messages are broadcast to all systems on a LAN subnet using the subnet broadcast address. You can specify that the DECNIS additionally sends RIP messages to specific systems by specifying them as IP sinks.

   Enter the IP address of the specific systems in the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name IP SINKS -
   _NCL> {a.b.c.d,...}
   ```

   where *a.b.c.d* is the IP address of the system to send messages to.

   If you want the DECNIS to send RIP messages only to the IP sinks, you must set the RIP MODE attribute to POINT-TO-POINT.

7. By default, the RIP MODE attribute is set to BROADCAST. In broadcast mode, RIP advertisements are sent to subnet broadcast addresses and any IP hosts specified as IP sinks. If you set the mode to POINT-TO-POINT, RIP advertisements will be sent only to the IP hosts specified as IP sinks. To set the RIP MODE, enter the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name RIP MODE mode
   ```

   where *mode* is one of the following:

   BROADCAST
   POINT-TO-POINT

8. Specify any IP sources.

   By default, the DECNIS accepts RIP advertisements from any RIP router in a subnet associated with the RIP instance. You can specify that only RIP advertisements from specific routers will be accepted.

   You must set the specific routers as IP sources by entering the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name IP SOURCES -
   _NCL> {a.b.c.d,...}
   ```

where *a.b.c.d* is the IP address of the router that advertisements will be received from.

Note

If you specify any IP sources, the DECNIS will only accept RIP advertisements from those sources. For example, if you have set up RIP sources, and the DECNIS has point-to-point RIP circuits, it will not accept RIP advertisements from the remote systems on these circuits unless you included their IP addresses in the set of IP sources.

9. Enable the RIP CONTROL PROTOCOL entity by entering the following command:

```
NCL> ENABLE
ROUTING CONTROL PROTOCOL control-prot-name
```

10.10.4 Using the DECNIS Text-Based Configurator to Set Up RIP

If you set up the RIP protocol using the DECNIS text-based configurator, the configurator sets up one instance of RIP for each RIP state that you specify: Receive only, Send and Receive, or Send only. The configurator sets up names and default values as shown in Table 10–1.

Table 10–1 RIP Configurator Defaults

Rip State	Receive Only, Send and Receive, Send Only
control-prot-name	RipRx (receive only), RipTxRx (Send and Receive), RipTx (send only)
IP Subnet	The addresses and masks of all subnets associated with each RIP circuit
Poisoned Reverse	True
IP sinks	None
RIP mode	Broadcast
IP sources	IP addresses of all RIP routers in the subnets associated with the RIP instance

To see how RIP has been set up, look at the master NCL script that is created when you exit the configurator.

The following is an example RIP section of an NCL script. In the example, a RIP instance has been created for a Send and Receive RIP state:

```
! Create a RIP Send and Receive Control protocol

create routing control protocol RipTxRx type IP RIP
set routing control protocol RipTxRx rip state Send and Receive
set routing control protocol RipTxRx ip sources { 1.6.1.1  -
     ,  1.4.1.2 }
set routing control protocol RipTxRx ip subnets {{addr= 1.6.1.0 -
    ,mask= 255.255.255.0 }  , {addr= 1.4.1.0 ,mask= 255.255.255.0}}
```

10.10.5 Using the clearVISN DECNIS Configurator to Set Up RIP

To set up RIP using the clearVISN DECNIS configurator, follow these steps:

1. On the Main Navigation window, click IP Routing.
2. On the IP tab page, tick the RIP box.
3. On the Circuits tab page, select a circuit that you want to use for RIP, click the "Run IP over this circuit" box, and supply an IP address.
4. Enter the required information on the RIP tab page.
5. Enter the required information on the RIP Subnets tab page.

10.11 Using the BGP Protocol

The Border Gateway Protocol (BGP) is an Exterior Gateway Protocol used to link autonomous systems. BGP routers in each autonomous system communicate network reachability information based on a set of policies established within each autonomous system.

10.11.1 BGP Neighbors

Routers that communicate with each other using BGP are known as BGP neighbors. There are two types of BGP neighbor:

- **Internal BGP neighbors** are located within the same autonomous system.
- **External BGP neighbors** belong to different autonomous systems, but share the same subnetwork.

BGP neighbors communicate using the TCP protocol.

10.11.2 BGP Policies

BGP enforces routing policies. Policies are sets of criteria that determine which routes BGP receives and propagates, and the preference assigned to them. BGP routes include information that enables a BGP router to make policy decisions; this information includes a list of all the autonomous systems traversed so far by the route.

On the DECNIS, you implement policies by setting up RECEIVE DESTINATION and PROPAGATE DESTINATION subentities of the ROUTING CONTROL PROTOCOL for BGP. Refer to Chapter 12 for details.

10.12 Configuring the DECNIS to use the BGP Protocol

This section describes how to configure the DECNIS to use BGP.

10.12.1 Requirements for Running BGP

In order for a DECNIS to communicate using BGP, it must be running the TCP protocol.

10.12.2 Requirements for BGP Communication Between Autonomous Systems

In order for external BGP neighbors to communicate, they must share the same subnetwork.

10.12.3 BGP Configuration Information

This section describes information you need before you start configuring BGP.

10.12.3.1 BGP and the ROUTING CONTROL PROTOCOL entity

On the DECNIS, BGP is controlled by the ROUTING CONTROL PROTOCOL, type BGP. You can only have one such entity on a DECNIS.

10.12.3.2 BGP and Route Propagation and Filtering

You must configure route propagation and filtering for BGP. If you do not do this, BGP cannot receive any routing updates, including BGP updates, nor can it propagate routes to external BGP neighbors or routers running Interior Gateway Protocols (IGPs), such as RIP or OSPF.

In order to configure route propagation and filtering, you set up RECEIVE DESTINATION and PROPAGATE DESTINATION entities. Refer to Section 10.12 and Chapter 12 for details.

10.12.3.3 Order of Enabling Entities

It is important that you enable the RECEIVE DESTINATION and PROPAGATE DESTINATION entities for BGP **before** you enable the ROUTING CONTROL PROTOCOL for BGP. Otherwise, the information in the RECEIVE and PROPAGATE DESTINATION entities may not take effect for a long time.

The reason for this is that BGP does not re-advertise routes periodically; it only re-advertises routes when there are changes to routing information. Thus, if you change the RECEIVE or PROPAGATE DESTINATION entities, the changes will not take effect until a new routing update is advertised. Typically, a routing update is only advertised when a router goes down, or the network configuration is changed.

If the RECEIVE and PROPAGATE DESTINATION entities are enabled after the ROUTING CONTROL PROTOCOL, the information in them will take effect immediately. This is because BGP will not start advertising routes until its ROUTING CONTROL PROTOCOL entity is enabled.

10.12.3.4 Number of BGP External Neighbors

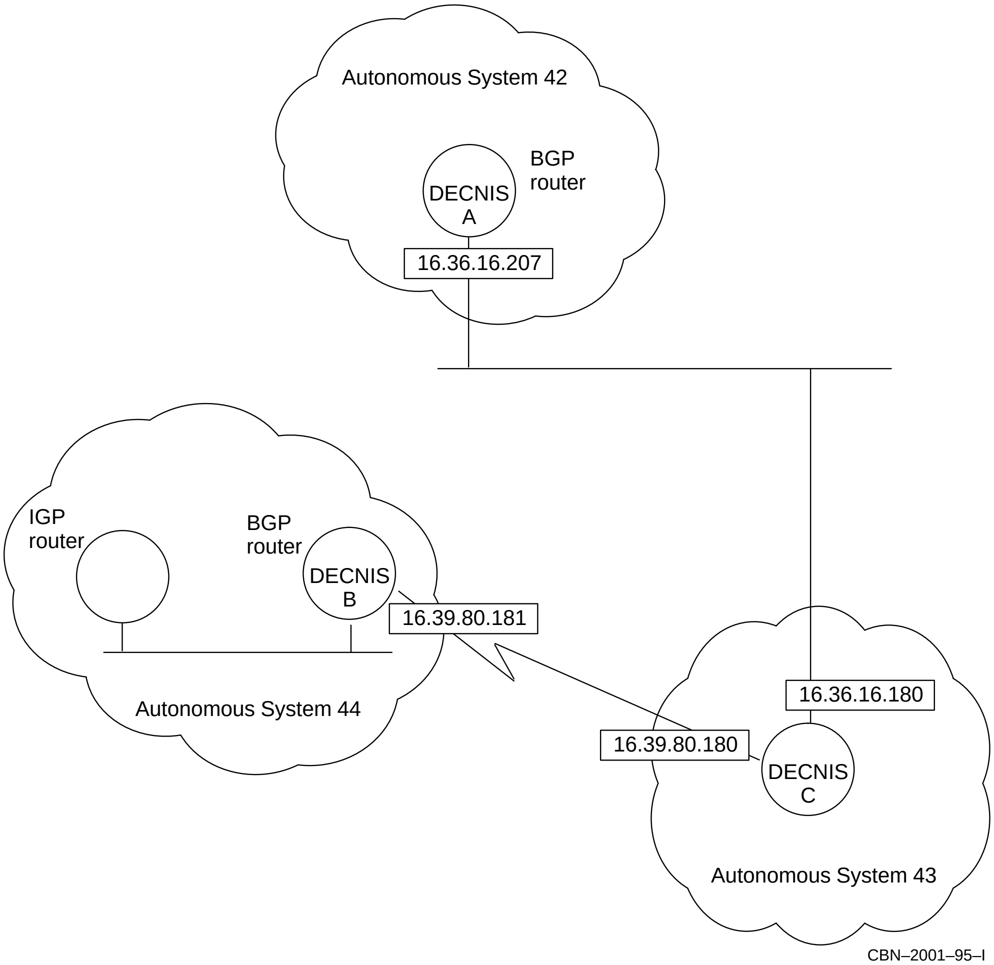
When you configure BGP, you specify the external BGP neighbors the DECNIS can communicate with. You are only allowed to specify up to 32 external BGP neighbors.

10.12.4 Example BGP Network

Figure 10–7 shows three autonomous systems. Each contains IP routers and hosts running a mixture of IGP routing protocols, such as IS–IS and RIP. In each autonomous system, one DECNIS running BGP has been designated to connect to another autonomous system.

Note that, in Figure 10–7, autonomous system 42 can communicate with autonomous system 44. This is because the BGP protocol allows the IP router to advertise IP hosts and routers that are reachable from within the router’s own autonomous system.

Figure 10–7 Communication Among Autonomous Systems Using the BGP Protocol



10.12.5 Procedure

This section describes how to configure a DECNIS as a BGP router. All example commands assume you are configuring DECNIS A in Figure 10–7.

Note that this section describes the minimum configuration required for a DECNIS to operate as a BGP router. Steps 8 to 12 describe how to configure a DECNIS to receive and propagate all routes. You can, if you wish, select the routes to be received and propagated, using various criteria; see Chapter 12 for details.

Follow these steps:

1. Create and enable TCP on the DECNIS. Note that this is done automatically by the DECNIS configurator (text-based or Windows). Alternatively, you can enter the following commands:

   ```
   NCL> CREATE TCP
   NCL> ENABLE TCP
   ```

2. Ensure that the Autonomous System Number has been specified by entering the following command:

   ```
   NCL> SHOW ROUTING AUTONOMOUS SYSTEM NUMBER
   ```

   If there is no Autonomous System Number, you must set it by entering the following command:

   ```
   NCL> SET ROUTING AUTONOMOUS SYSTEM NUMBER as-number
   ```

   where *as-number* is the autonomous system number.

3. Create a ROUTING CONTROL PROTOCOL entity of type BGP on the DECNIS:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-prot-name TYPE BGP
   ```

   where *control-prot-name* is the ROUTING CONTROL PROTOCOL name. Note that you can have only one CONTROL PROTOCOL for BGP per DECNIS. It is recommended that you use the name bgp.

4. Create a NEIGHBOR entity for each autonomous system that is reachable from this DECNIS using BGP. You are only allowed to specify up to 32 external BGP neighbors.

   Enter the following command:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-prot-name NEIGHBOR neighbor
   ```

   where *neighbor* is the name of a NEIGHBOR entity representing an external BGP neighbor.

   For example:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL bgp NEIGHBOR sys43
   ```

5. Specify the autonomous system number of the NEIGHBOR entity:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> NEIGHBOR neighbor AUTONOMOUS SYSTEM as-number
   ```

where *as-number* is the autonomous system number. For example:

```
NCL> SET ROUTING CONTROL PROTOCOL bgp NEIGHBOR sys43 -
_NCL> AUTONOMOUS SYSTEM 43
```

6. Specify the IP address of the external BGP neighbor. In the example, this would be the IP interface address of DECNIS C in autonomous system 43. For example:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp NEIGHBOR sys43 -
   _NCL> IP ADDRESS 16.36.16.180
   ```

   Note the following:

   - The DECNIS must be on the same subnet as the one specified in this IP address.
   - The DECNIS circuit connecting to the subnet must be enabled before you enable the NEIGHBOR entity.

7. Create a PROPAGATE DESTINATION entity:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> PROPAGATE DESTINATION name FILTER ACTION PASS
   ```

   where:

name	is a unique name to identify the PROPAGATE DESTINATION entity.
PASS	is the filter action, which indicates that the routes defined by this entity should be propagated.

8. Specify the address match type:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> PROPAGATE DESTINATION name ADDRESS MATCH TYPE PREFIX
   ```

9. Create a RECEIVE DESTINATION entity:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> RECEIVE DESTINATION name FILTER ACTION PASS
   ```

   where:

name	is a unique name to identify the RECEIVE DESTINATION entity.
PASS	is the filter action, which indicates that the routes defined by this entity should be received.

10. Set up the address match type for the RECEIVE DESTINATION:

    ```
    NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
    _NCL> RECEIVE DESTINATION name ADDRESS MATCH TYPE PREFIX
    ```

11. Enable the RECEIVE and PROPAGATE DESTINATION entities:

```
NCL> ENABLE ROUTING CONTROL PROTOCOL control-prot-name -
_NCL> PROPAGATE DESTINATION name
NCL> ENABLE ROUTING CONTROL PROTOCOL control-prot-name -
_NCL> RECEIVE DESTINATION name
```

12. Enable BGP routing on the DECNIS by enabling the CONTROL PROTOCOL. For example:

```
NCL> ENABLE ROUTING CONTROL PROTOCOL bgp
```

13. Enable BGP routing to the neighboring system; for example:

```
NCL> ENABLE ROUTING CONTROL PROTOCOL bgp NEIGHBOR sys43
```

10.12.5.1 Mapping a Neighbor to a Circuit

How does BGP know which circuits are being used for BGP?

When you create ROUTING CONTROL PROTOCOL NEIGHBOR for BGP, you identify an autonomous system that BGP will connect to, and the IP address of the external BGP neighbor for that autonomous system. When you enable the NEIGHBOR, the DECNIS checks that the NEIGHBOR's subnet, as specified in the IP address, is the same as that of a currently enabled circuit on the DECNIS. This circuit is the one that will be used for connections to the specified autonomous system.

For this reason, you should make sure that you do in fact have an enabled DECNIS circuit connecting to the subnet specified in the NEIGHBOR entity.

10.13 Configuring the DECNIS to Use the EGP Protocol

10.13.1 Introduction

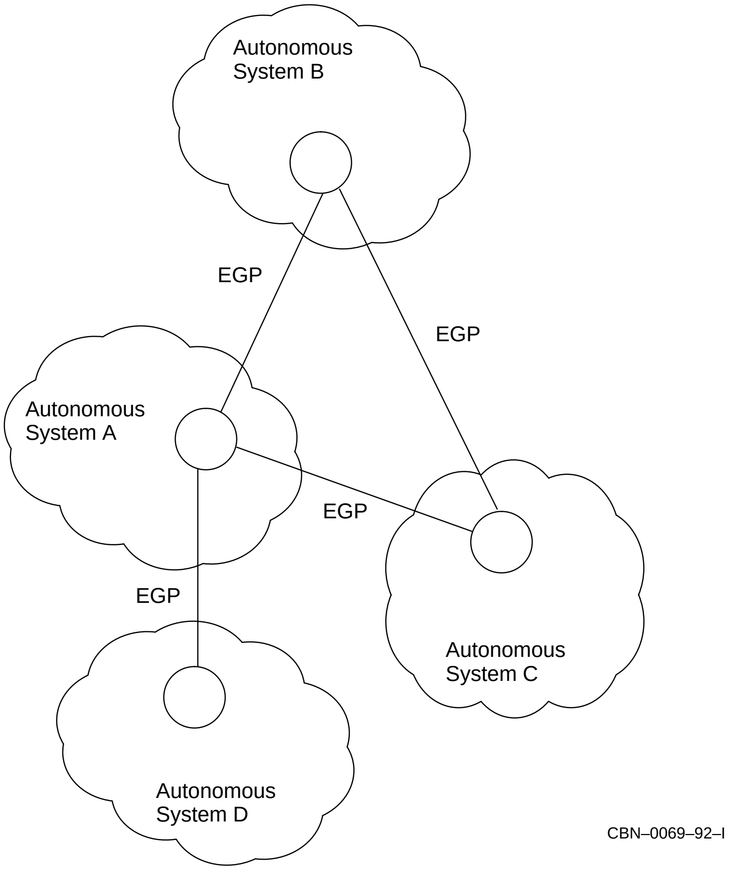
The EGP protocol is used to exchange IP routes between autonomous systems. Figure 10–8 shows four autonomous systems, each of which can contain IP routers and hosts running a mixture of IGP routing protocols, such as IS–IS and RIP. In each autonomous system, one IP router has been designated to propagate routes to other autonomous systems. The designated IP router in each autonomous system communicates with the designated routers in other autonomous systems using EGP.

Note that, in Figure 10–8, autonomous systems B and C cannot communicate with autonomous system D. This is because the EGP protocol only allows an IP router to advertise IP hosts and routers that are reachable from within the router's own autonomous system.

10.13.2 Requirements for Running EGP

The DECNIS must be configured as a Level 2 router in order to run EGP.

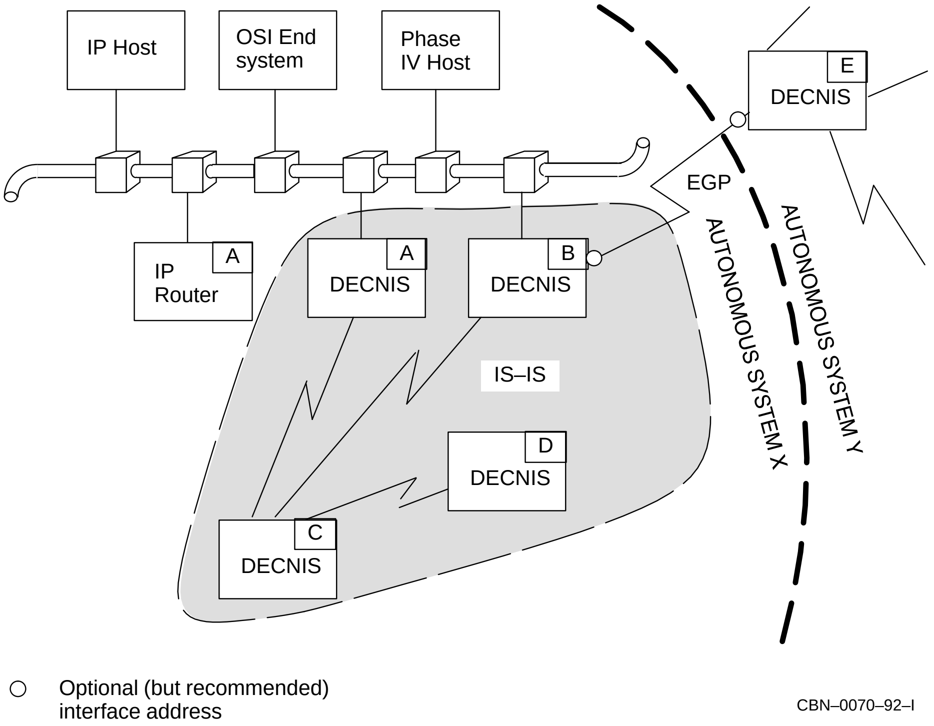
Figure 10–8 Communication Among Autonomous Systems Using the EGP Protocol



10.13.3 Example Procedure

An example of a DECNIS running EGP is shown in Figure 10–9. Follow these steps to enable DECNIS B in Figure 10–9 to run the EGP protocol:

Figure 10–9 Using EGP in an IP Network



1. Ensure that the Autonomous System Number of DECNIS B has been set:

   ```
   NCL> SHOW ROUTING AUTONOMOUS SYSTEM NUMBER
   ```

   Refer to Section 10.3.2 for more information about the autonomous system number.

   If the Autonomous System Number has not been set, set it using the DECNIS text-based configurator, and reboot the DECNIS.

2. Assign an IP address and subnet mask to the EGP interface as shown in Figure 10–9: see Section 10.5.

3. If the routing circuit is an X.25 static circuit or uses an HDLC or DDCMP data link, specify the IP address of the adjacent system:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit-name
   NCL> SET ROUTING CIRCUIT circuit-name NEIGHBOR IP ADDRESS = -
   _NCL> a.b.c.d
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

   where *a.b.c.d* is the IP address of DECNIS E.

4. Create the EGP GROUP entity for autonomous system Y:

   ```
   NCL> CREATE ROUTING EGP GROUP EGP-group-name
   ```

5. Specify the Autonomous System Number for Autonomous System Y:

   ```
   NCL> SET ROUTING EGP GROUP EGP-group-name -
   _NCL> AUTONOMOUS SYSTEM NUMBER n
   ```

   where *n* is a decimal number between 0 and 65,535. It is assigned to the autonomous system by the central authority for assigning IP network addresses.

   Note that the autonomous system number you enter here must not be the same as that in step 1. That is, you cannot use EGP on the DECNIS in an intra-domain configuration.

6. Enable the EGP GROUP entity:

   ```
   NCL> ENABLE ROUTING EGP GROUP EGP-group-name
   ```

7. Now create an EGP NEIGHBOR entity for each router in Autonomous System Y with which DECNIS B can communicate. In the configuration shown, there is only one router: DECNIS E. Enter the command:

   ```
   NCL> CREATE ROUTING EGP GROUP EGP-group-name -
   _NCL> EGP NEIGHBOR EGP-neighbor-name IP ADDRESS IP-address, -
   _NCL> CIRCUIT circuit-name
   ```

   where:

EGP-neighbor-name	is a name you can use to identify this router.
IP-address	is the IP address of DECNIS E.
circuit-name	is the name of the routing circuit, defined on DECNIS B, connecting to DECNIS E.

8. Enable the EGP NEIGHBOR entity:

```
NCL> ENABLE ROUTING EGP GROUP EGP-group-name -
_NCL> EGP NEIGHBOR EGP-neighbor-name
```

10.13.4 EGP Route Propagation and Filtering

If you want to propagate or filter EGP routes, you must have a CONTROL PROTOCOL entity for each autonomous system to which you wish to apply the propagation or filtering. Each CONTROL PROTOCOL entity is known as an EGP instance.

To set up a CONTROL PROTOCOL entity, follow these steps:

1. Create the entity by entering the following command:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-prot-name TYPE EGP
   ```

   where *control-prot-name* is the name you want to give to this entity. For example:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL EGP-1 TYPE EGP
   ```

2. Associate the CONTROL PROTOCOL with the EGP GROUP entity of the remote autonomous system by entering the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL EGP-1 AUTONOMOUS SYSTEM NUMBER n
   ```

   where *n* is the Autonomous System Number for the EGP GROUP. For example, in Section 10.13.3, this is the number that you specify in step 5.

3. Enable the EGP CONTROL PROTOCOL entity by entering the following command:

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL control-prot-name
   ```

For more information about route propagation and filtering, refer to Chapter 11.

10.13.5 Using the DECNIS Text-Based Configurator to Set Up EGP Route Propagation

You can set up the EGP protocol using the DECNIS text-based configurator. Note that EGP is not supported in the clearVISN DECNIS configurator.

If you specify that you want to propagate EGP routes, the DECNIS text-based configurator sets up one Control Protocol entity for each autonomous system associated with an EGP group.

For example, if you have an Autonomous System Number of 111 assigned to an EGP group, then the configurator will use the following NCL commands to set up an EGP Control Protocol:

```
create routing control protocol EGP111 type EGP
set routing control protocol EGP111 autonomous system number 111
```

To find out how EGP has been set up, look at the master NCL script created by the DECNIS text-based configurator.

To change any of the EGP settings, edit the user NCL script files and reboot the DECNIS. For details about the user NCL script files, refer to Section 1.5.6.

10.14 Static Routes

10.14.1 Introduction

Static IP routes are similar to static OSI routes (see Section 16.15).

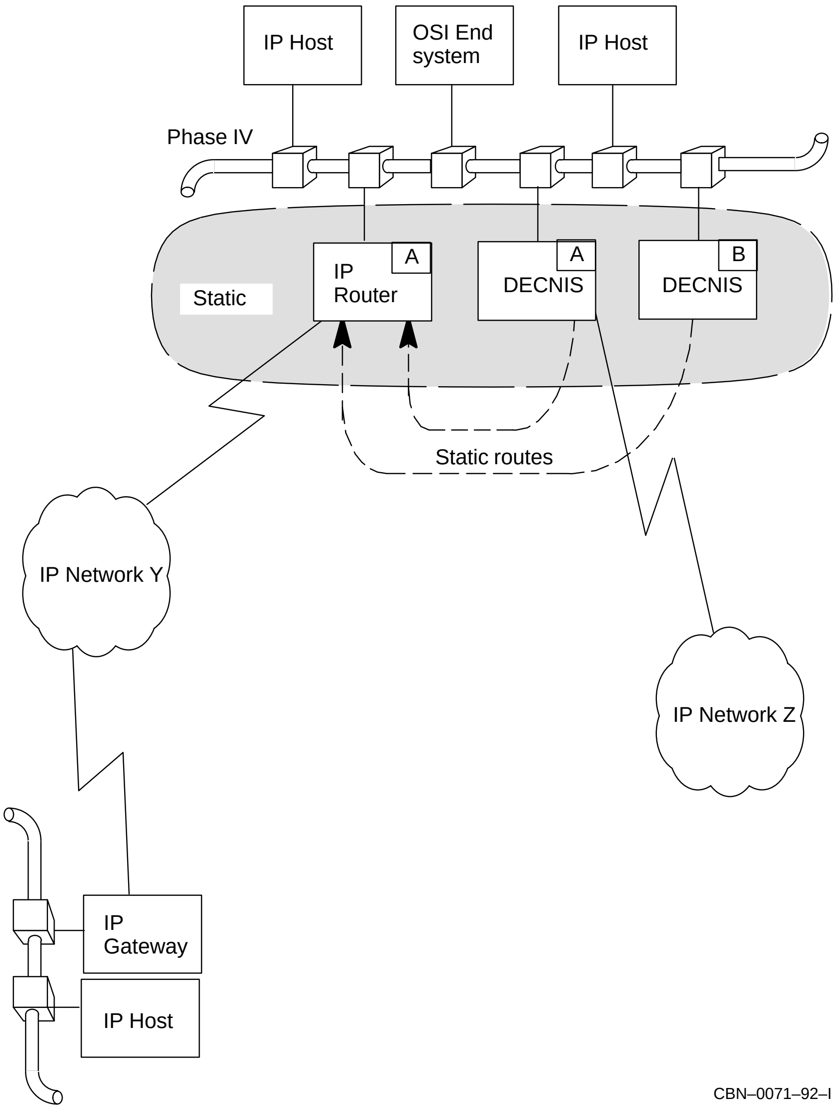
This section explains how to configure the DECNIS so that it can send IP packets to IP routers with which it does not exchange routing information, for example, if the circuit type is X25 DA.

10.14.2 Example

In the example shown in Figure 10–10, DECNIS systems A and B run RIP but IP Router A does not. Therefore, if hosts or routers in IP Network Z wish to send packets to IP Network Y, DECNIS A must be informed explicitly about IP Router A. This is done by setting up an IP reachable address on DECNIS A.

Note that if you want to send packets to IP Network Z from IP Network Y, you must also set up a static route on IP Router A. Refer to the documentation for IP Router A for details of how to do this.

Figure 10–10 Setting Up a Static Route



10.14.3 Setting Up a Static Route

To set up a static route on the DECNIS, follow these steps:

1. If the routing circuit on which the reachable address is to be set up is an X.25 static circuit or uses an HDLC or DDCMP data link, specify the IP address of the adjacent system:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit-name
   NCL> SET ROUTING CIRCUIT circuit-name NEIGHBOR IP ADDRESS = -
   _NCL> i.j.k.l
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

   For example, in Figure 10–10, *i.j.k.l* is the IP address of IP Router A.

2. Set up the reachable address for the IP network to be reached by this static route (Network Y in Figure 10–10), specifying the IP address of the network:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name -
   _NCL> IP REACHABLE ADDRESS address-name -
   _NCL> DESTINATION = {ADDRESS=a.b.c.d, MASK=e.f.g.h}
   ```

   where *a.b.c.d* is the address of Network Y, and *e.f.g.h* is its subnet mask.

3. Specify the address of the router used to reach this network:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> IP REACHABLE ADDRESS address-name NEXT HOP i.j.k.l
   ```

   where *i.j.k.l* is the IP address of IP Router A in Figure 10–10.

   Note that if the circuit type is X.25 dynamically assigned, you omit this step.

4. In this example, the routing circuit is a CSMA/CD circuit. If the routing circuit is an X.25 dynamically assigned circuit, you must specify the DTE address(es) of the IP router(s) that can be used to reach this network.

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> IP REACHABLE ADDRESS address-name -
   _NCL> DTE ADDRESSES {DTE-address1, DTE-address2,...}
   ```

5. Enable the reachable address:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name IP REACHABLE ADDRESS address-name
   ```

10.14.4 Setting the Preference of Static Routes

By default, static routes have a preference of 10. This is higher than the preference for any other routes except local routes.

You may wish to reduce this preference, so that another protocol will be preferred. There are two ways to do this:

- By using route filtering to change the preference of groups of static routes, or all static routes, as described in Section 11.5.4.
- By changing the preference for an individual IP reachable address, as described in Section 10.14.4.1.

10.14.4.1 Changing the Preference of an Individual IP Reachable Address

To change the preference for an individual IP reachable address, enter the following command:

```
NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> IP REACHABLE ADDRESS address-name PREFERENCE n
```

where *n* is the value of the new preference.

10.15 Default Routes

10.15.1 Definition

A default route is a method by which an IP router can route packets if there is no entry for the destination in its routing tables.

If an IP router receives a packet for a destination for which there is no entry in its routing tables, it will forward the packet to an IP router that is advertising a default route.

10.15.2 Types of Default Route

The DECNIS can use the following types of default route:

- Level 2 router default route

  If the DECNIS is an IS–IS Level 1 router, it automatically inserts a route to the nearest IS–IS Level 2 router into its routing table. If the DECNIS receives an IP packet for a destination not in its routing table, it will forward the packet to the nearest IS–IS Level 2 router.
- RIP, IS–IS, OSPF, and EGP default routes

If the DECNIS receives a default route from any routers running RIP, IS–IS, OSPF or EGP, it inserts the route into its routing table. If it then receives an IP packet for a destination not listed in its routing table, it sends the packet to the router that advertised the default route.

- Static default routes

  On the DECNIS you can set up a static default route to any adjacent system. This means that if the DECNIS receives a packet for a destination which is not listed in its routing table, it will send it to the adjacent system.

  Section 10.15.3 describes how to create a static default route.

If the DECNIS has several default routes to choose from, it will use preferences and metrics to determine which route to use.

You can prevent the DECNIS from listening to default routes using route filtering; see Section 11.4. The DECNIS can generate a default route using route propagation; see Section 11.7.4.

10.15.3 Creating a Static Default Route

IP Reachable Address for Default Route

To set up a static default route, you must set up an IP reachable address on the circuit on which unknown destination address packets will be forwarded. Section 10.14 describes IP reachable addresses.

You must specify the destination of this reachable address as an address of 0.0.0.0 and a mask of 0.0.0.0:

Procedure

Create a static default route as follows:

1. If the routing circuit on which the packets are to be forwarded is a point-to-point circuit, specify the IP address of the adjacent system:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name NEIGHBOR IP ADDRESS = -
   _NCL> i.j.k.l
   ```

2. Set up a reachable address on the circuit, specifying a destination address and subnet mask of 0.0.0.0:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name -
   _NCL> IP REACHABLE ADDRESS address-name -
   _NCL> DESTINATION = {ADDRESS=0.0.0.0, MASK=0.0.0.0}
   ```

3. Specify the address of the router used to reach this network:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> IP REACHABLE ADDRESS address-name NEXT HOP i.j.k.l
   ```

   where *i.j.k.l* is the IP address of an adjacent IP router.

   Note that if the circuit type is X.25 dynamically assigned, you omit this step.

   If the routing circuit is an X.25 dynamically assigned circuit, you must specify the DTE address(es) of the IP router(s) that can be used to reach this network.

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> IP REACHABLE ADDRESS address-name -
   _NCL> DTE ADDRESSES {DTE-address1, DTE-address2,...}
   ```

4. Enable the reachable address:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name -
   _NCL> IP REACHABLE ADDRESS address-name
   ```

10.15.4 Changing the Preference of Static Default Routes

A static default route, like other static routes, has a higher preference than any other type of route, except a local route. This means that it is possible for default static routes to mask other routes. To prevent this, you can either use route filtering, as described in Section 11.5.4 or reduce the preference of individual static routes, as described in Section 10.14.4.

10.16 Summary IP Routes in IS–IS Level 2 Routers

10.16.1 Introduction

You can reduce network traffic by using summary routes. To do this, you configure your network so that each subnet is in a different DECnet/OSI area, and set a summary address routing characteristic on each Level 2 router. In this way, only summary IP address information is sent between areas.

Note that summary addresses do not apply to BGP routing. However, you can summarize BGP routes by using BGP route aggregation; see Section 10.19 for details.

10.16.2 Example

Figure 10–11 shows an IP autonomous system (address 17.0.0.0) in which all subnets of the form 17.133 are one area, 17.22 are in another, and 17.42 are in a third.

By setting up summary information on DECNIS A, it will only send information about subnet 17.133.0.0, rather than information about all the individual IP subnets 17.133.43.0, 17.133.57.0, and 17.133.5.0.

Similarly, the Level 2 router in Area 2 can be configured with a summary address of 17.22.0.0, so that it only sends information about subnet 17.22.0.0, rather than information about all the individual IP subnets within the area.

All the Level 2 routers in an area must be configured with the same set of summary addresses.

10.16.3 Requirement for Setting Up Summary IP Information

You can only set up summary IP addresses on the DECNIS if it is configured to run link state routing at Level 2. This restriction does not apply to OSPF routers.

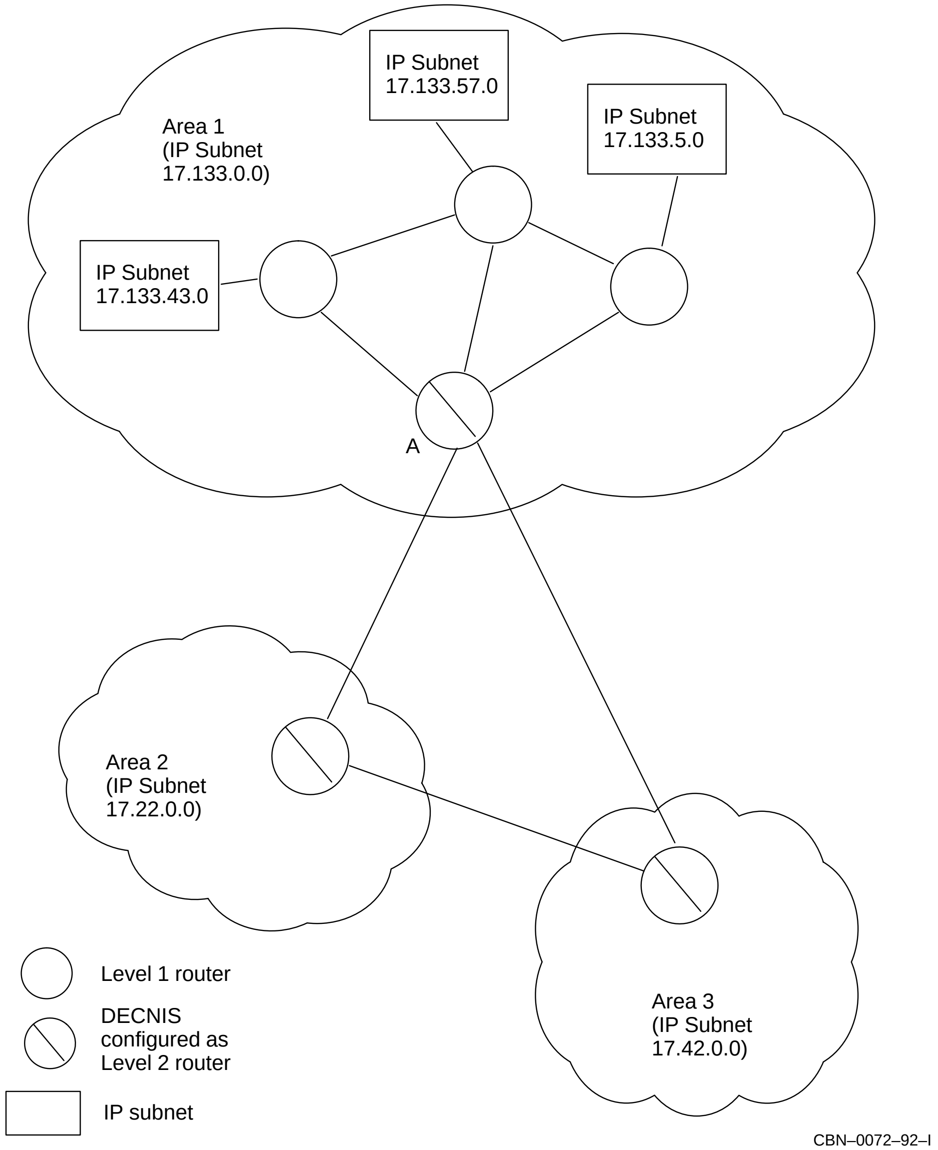
10.16.4 Setting Up Summary IS–IS Addresses on the DECNIS

On DECNIS A in Figure 10–11, you would set up summary address information as follows:

```
NCL> ADD ROUTING SUMMARY ADDRESS {[SUBNET= -
_NCL> [ADDRESS=17.133.0.0, MASK=255.255.0.0], METRIC=5]}
```

where the METRIC value is the metric to be used when abbreviating the route at Level 2.

Figure 10–11 Using Summary Routes



10.17 Setting Up Summary OSPF Addresses on the DECNIS

OSPF areas are identified to the IP network by their addresses. These consist of a set of address ranges, listed in pairs (address, mask). Many variable sized subnets may be defined within an OSPF area, but all can be identified by a single address range, using one address with an internet mask.

This allows an Area Border Router to summarize the Area contents for distribution to the backbone, by advertising a single route for each address range. The route cost is taken as the minimum cost to any of the networks falling within the specified range.

For example, you could configure your network so that each subnet is in a different OSPF area, and set a summary address routing characteristic on each Area Border router. Only summary IP address information would be sent between areas.

10.18 Setting Up Summary Routes for an OSPF Area

10.18.1 Introduction

Variable-length IP subnet masking allows flexibility in the definition of address bits in the subnet mask. For example, if you have an OSPF network with many hosts and few subnetworks, you could define subnet masks to identify more hosts than subnetworks.

OSPF attaches an IP address mask to each advertised route. The mask indicates the range of addresses being described by the particular route. This allows a single network to be subdivided into variable-sized subnetworks, all addressed by a single address range. Area Border Routers then summarize the Area addresses by only advertising a single route for each address range. The route costs are the minimum costs to any of the subnetworks falling in the specified range.

10.18.2 Example

The following example shows how to configure Summary routes for an Area. The example refers to Figure 10–11.

You wish to create a single Summary route for the networks in the range 17.133.0.0 to 17.133.255.255. Enter the following command:

```
NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-1 -
_NCL> OSPF AREA RANGE { (ADDRESS = 17.133.0.0, MASK = 255.255.0.0) }
```

There is no need to supply a METRIC value; OSPF will advertise a cost which is the minimum cost to any of the networks falling into the specified range.

10.19 BGP Route Aggregation

10.19.1 Introduction

BGP route aggregation allows you to minimise the amount of IP routing information on the network, as the use of summary routes does for other routing protocols. As with summary routes, BGP route aggregation lets you propagate multiple IP routes as a single update rather than individual updates.

For example, the subnet routes 16.36.16, 16.36.20 and 16.36.8 could be replaced by a single route to 16.36.

10.19.2 General Method for Setting Up BGP Aggregation

BGP route aggregation works by using IP domains. An IP domain is a set of IP address/mask pairs, specified in a DOMAIN subentity of the ROUTING NETWORK PROTOCOL IP entity.

Each address/mask pair identifies a unique set of address ranges. Overlapping ranges are not allowed for BGP route aggregation.

To set up BGP route aggregation, you set up an IP domain, with an associated set of address/mask pairs, and then specify that BGP route aggregation applies to that domain.

Aggregation assumes that the network topology is organised hierarchically and that the addresses reflect this hierarchy. Where exceptions occur, they can be handled by creating more specific domains. For example, if 16.36.20 lies in a different direction to 16.36.16 and 16.36.8 then three domains could be created which would omit 16.36.20 from the aggregation.

10.19.2.1 Aggregation and the Routing Update Process

During the routing update process, each IP address for propagation is compared with each address/mask pair in the IP domains used for aggregation. If the address is found in a domain, it forms part of the aggregate routing information about that domain.

10.19.3 Example Procedure

The following example shows how to configure the DECNIS to allow BGP route aggregation for IP addresses that lie between 10.0.0.0 and 11.0.0.0. Follow these steps:

1. Disable the routing circuit corresponding to the IP interface:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit_name
   ```

2. Create the domain:

   ```
   NCL> CREATE ROUTING NETWORK PROTOCOL IP DOMAIN domain_name
   ```

   where *domain_name* is a name you create to identify the domain.

3. Set up a pair of IP addresses and subnet masks that define the domain IP address ranges:

   ```
   NCL> SET ROUTING NETWORK PROTOCOL IP DOMAIN domain_name IP ADDRESS -
   _NCL> {[addr=10.0.0.0, mask=255.0.0.0],[addr=11.0.0.0, mask=255.0.0.0]}
   ```

4. Set the domain to be used by BGP for route aggregation:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name BGP AGGREGATION -
   _NCL> {ROUTING NETWORK PROTOCOL IP DOMAIN domain_name}
   ```

5. Enable the routing circuit corresponding to the BGP interface:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit_name
   ```

10.20 Using All Subnets Broadcast

10.20.1 Introduction

By default, if an IP interface on the DECNIS receives an IP packet addressed to all the subnets in its IP network, none of the other IP interfaces on the DECNIS will forward it, and the packet is dropped.

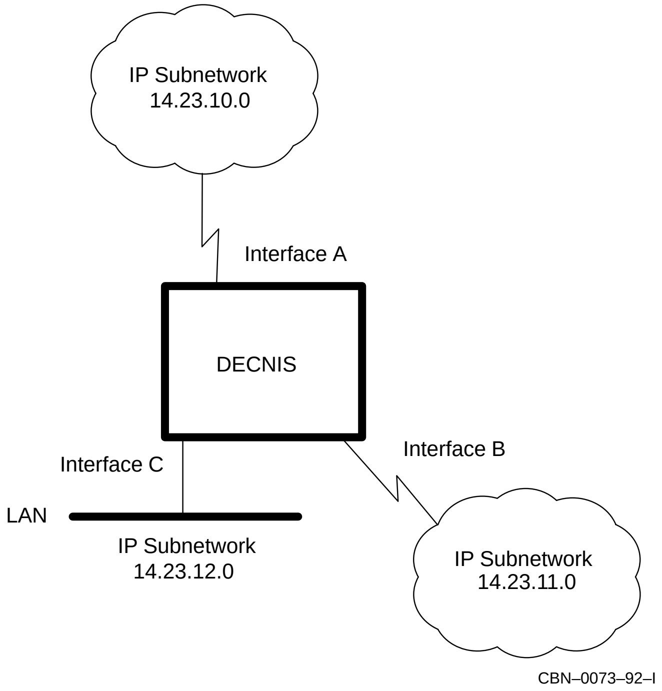
However, you can configure an interface on the DECNIS to forward these broadcast packets. The packets are then forwarded to the adjacent system(s) on that interface. This is known as All Subnets Broadcast, and is described in detail in RFC 922.

10.20.2 Example

For example, consider Figure 10–12. The IP network shown has a network address of 14.23.0.0. It is divided into three subnetworks, with the subnetwork addresses shown.

In this example, Interface A receives broadcast packets. These have the address 14.23.255.255. By default, these packets will be dropped. However, you can configure Interface B and/or Interface C to forward these packets to the IP subnetworks to which they connect.

Figure 10–12   All Subnets Broadcast



Multiple Copies of Packets

The DECNIS uses a reverse path lookup mechanism to avoid packet loops with All Subnet Broadcast. However, multiple copies of packets may be received if there are multiple paths with similar costs.

10.20.3 Data Link Broadcast

Note that the DECNIS will never forward a packet that it receives by data link broadcast. For example, in Figure 10–12, if a system on the LAN broadcasts a packet to all systems on the LAN, the DECNIS will not forward it, even if All Subnets Broadcast is enabled on Interface A or Interface B.

10.20.4 Procedure

Force an IP interface to forward broadcast packets by entering the following command:

```
NCL> SET ROUTING CIRCUIT circuit-name ALL SUBNETS BROADCAST ON
```

10.21 Using UDP Broadcast Forwarding

10.21.1 Introduction

By default, if a DECNIS receives a UDP packet broadcast on one subnet, it does not forward the packet to other subnets or routing circuits.

However, you can configure the DECNIS to receive UDP packets broadcast on one circuit and forward them to one or more UDP servers. The UDP servers may be on other circuits, or on the same circuit but in different subnets. This is known as UDP Broadcast Forwarding.

UDP Broadcast Forwarding is useful in situations where IP hosts on different circuits are all members of the same destination subnet, or the server process for the UDP packet is on an IP host in a different subnet from the destination.

10.21.2 UDP Broadcast Forwarding to a Single Server

Example

Figure 10–13 shows an example network with UDP Broadcast Forwarding. LAN-A has subnet 192.3.4.0 with the mask 255.255.255.0. Hosts on LAN-A broadcast UDP packets to ports 2501 and 2502. The destination address of the broadcast packets is either the network broadcast address 192.3.4.255 or the limited broadcast address 255.255.255.

You can configure the DECNIS to forward the UDP packets to the UDP server on the IP host at 16.1.2.3.

10.21.3 General Procedure

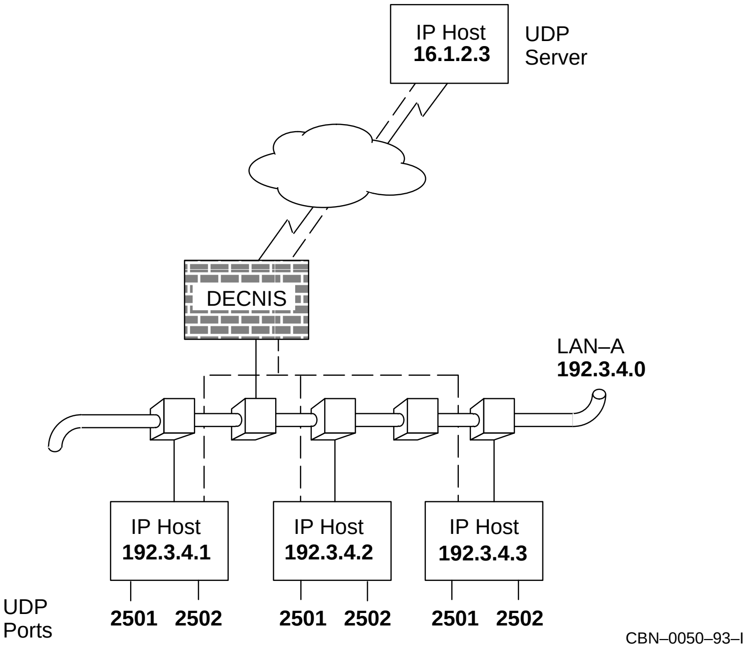
To configure the DECNIS to forward UDP broadcast packets to a UDP server, enter the following commands:

```
NCL> CREATE ROUTING CIRCUIT circuit-name UDP SERVER server-name -
_NCL> ADDRESS ip-address
NCL> SET ROUTING CIRCUIT circuit-name UDP SERVER server-name PORTS {port-list}
NCL> ENABLE ROUTING CIRCUIT circuit-name UDP SERVER server-name
```

where:

server-name	is the name of the UDP SERVER entity.
ip-address	is the IP address of the UDP server to which you will forward UDP packets.
{port-list}	is the list of UDP ports for which packets are to be forwarded.

Figure 10–13 DECNIS Using a Single UDP Server



UDP broadcast packets destined for any of the ports listed will be forwarded to the IP address given.

Example Procedure

To create a UDP SERVER entity for the example in Figure 10–13, enter the following commands:

```
NCL> CREATE ROUTING CIRCUIT LAN-A UDP SERVER applserver ADDRESS 16.1.2.3
NCL> SET ROUTING CIRCUIT LAN-A UDP SERVER applserver PORTS {2501,2502}
NCL> ENABLE ROUTING CIRCUIT LAN-A UDP SERVER applserver
```

Once the UDP SERVER has been enabled, any broadcast UDP packet for port 2501 or 2502 received over LAN-A will be forwarded to the server with the destination address set to 16.1.2.3. The packet is forwarded normally, using the routing table to determine over which circuit it is sent.

10.21.4 UDP Broadcast Forwarding to Multiple Servers

You may want UDP packets received over a circuit for a particular port to be forwarded to more than one server. To do this, you should configure more than one UDP SERVER entity on the circuit. Each UDP SERVER will have the particular port in its PORTS list.

Note that the name of a UDP SERVER entity and its ADDRESS must be unique for the circuit. Two UDP SERVER entities can only have the same name or ADDRESS value if they are on different circuits.

10.21.5 UDP Broadcast Forwarding to a Broadcast Address

It is possible to specify the ADDRESS of a UDP SERVER as an IP broadcast address rather than a host address. The DECNIS will then re-broadcast any UDP broadcast packet received for the specified port or ports. The packet will be re-broadcast over any interface for which the IP address specified in ADDRESS is an IP broadcast address. The destination address of the re-broadcast packets will be that specified by ADDRESS.

Example

As an example, consider the network shown in Figure 10–14.

The DECNIS is attached to four LANs:

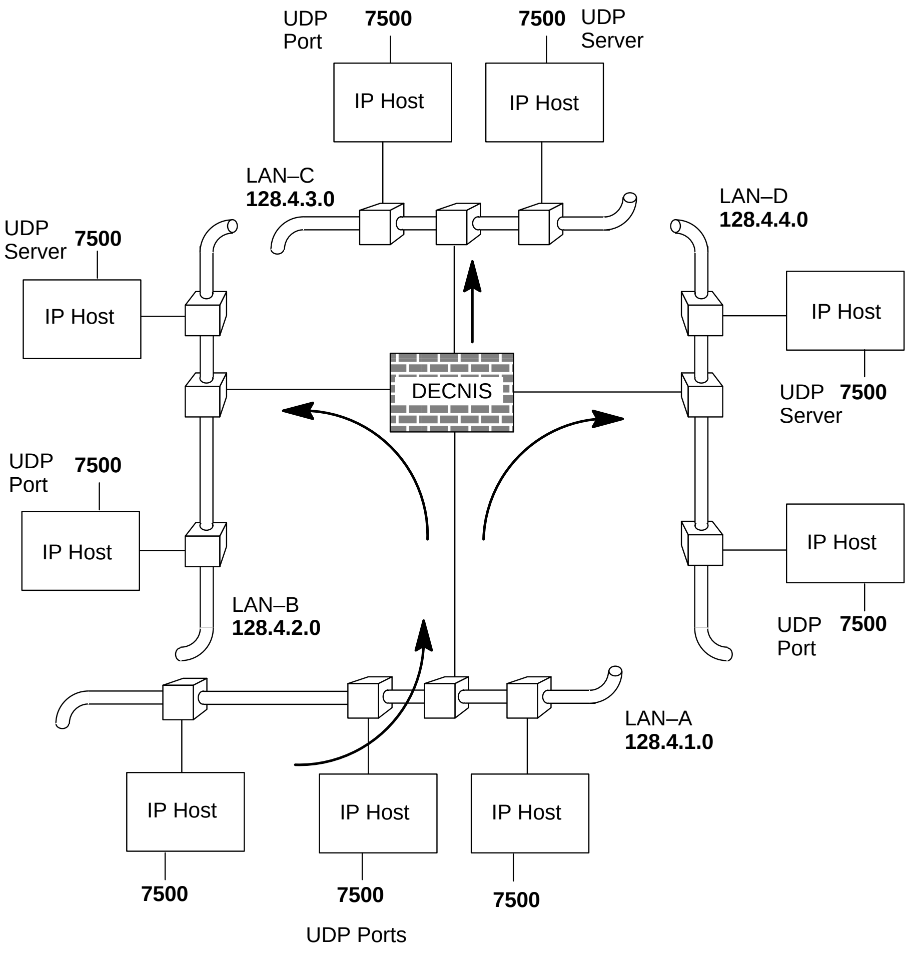
LAN-A has subnet address 128.4.1.0, mask 255.255.255.0
LAN-B has subnet address 128.4.2.0, mask 255.255.255.0
LAN-C has subnet address 128.4.3.0, mask 255.255.255.0
LAN-D has subnet address 128.4.4.0, mask 255.255.255.0

Assume that a host on LAN-A is broadcasting UDP packets for port 7500 to its subnet broadcast address 128.4.1.255. These packets need to be forwarded by the DECNIS onto the other three LANs. To do this, create a UDP SERVER entity with the ADDRESS set to the network broadcast address 128.4.255.255 using these commands:

```
NCL> CREATE ROUTING CIRCUIT LAN-A UDP SERVER p7500serv ADDRESS 128.4.255.255
NCL> SET ROUTING CIRCUIT LAN-A UDP SERVER p7500serv PORTS {7500}
NCL> ENABLE ROUTING CIRCUIT LAN-A UDP SERVER p7500serv
```

When the DECNIS receives the packets from LAN-A it will re-broadcast them onto LAN-B, LAN-C and LAN-D since they are all in network 128.4.0.0. The destination address of the packets will be 128.4.255.255.

Figure 10–14 DECNIS Using a Broadcast Address



10.22 Setting Up Permanent ARP Cache Entries

This section describes how to set up a permanent ARP cache entry for an individual IP address on the DECNIS.

10.22.1 ARP (Address Resolution Protocol)

IP hosts on a LAN normally communicate using ARP (Address Resolution Protocol). An IP host broadcasts an ARP request message on the LAN, specifying the IP address of the destination host. The destination host then responds, supplying its physical address in the response packet. The source IP host can then send IP packets to the destination using the destination's physical address.

Systems using ARP keep a cache of recently learned mappings of IP address to physical address, to reduce the number of times that they have to use ARP broadcasts.

10.22.2 Permanent ARP Cache Entries

You can create permanent ARP cache entries on the DECNIS for individual IP addresses. If an ARP cache entry for the IP address already exists, it will have its physical address set to the one supplied for the permanent cache entry, and will be marked as a permanent entry.

Once you have created a permanent ARP cache entry, you can only delete it using network management; it will not be automatically updated or cleared.

10.22.3 Creating a Permanent ARP Cache Entry

To create a permanent ARP cache entry for an IP address, enter the following command:

```
NCL> CREATE ROUTING CIRCUIT circuit-name -
_NCL> IP ADDRESS TRANSLATION a.b.c.d -
_NCL> LAN ADDRESS nn-nn-nn-nn-nn-nn-nn
```

where:

a.b.c.d is the IP address of the permanent entry. This is the identifier of an IP ADDRESS TRANSLATION entity.

nn-nn-nn-nn-nn-nn-nn is the physical address to associate with the IP address.

10.22.4 Deleting a Permanent ARP Cache Entry

To delete a permanent ARP cache entry for an IP address, enter the following command:

```
NCL> DELETE ROUTING CIRCUIT circuit-name -
_NCL> IP ADDRESS TRANSLATION a.b.c.d
```

where *a.b.c.d* is the IP address of this IP Address Translation entity.

10.23 Using Proxy ARP

10.23.1 Proxy ARP: Definition

A DECNIS configured to use proxy ARP on a LAN can respond to an ARP request message on behalf of a system that is not on the LAN. The source IP host then sends IP packets to the hardware address of the DECNIS. The DECNIS forwards these IP packets on to the real destination IP host.

10.23.2 How Proxy ARP Operates on the DECNIS

To use proxy ARP, you must enable it on the relevant LAN interface(s). When the DECNIS receives an ARP request message on one of its LAN interfaces, it first checks if the destination IP address is reachable on this same interface. If it is not, and proxy ARP is enabled on that interface, it will check its routing table to see if there is an entry for the destination address.

If it does have an entry for the destination address, it will respond to the ARP request message, and forward IP packets from the source IP host to the real destination on the relevant interface.

10.23.3 Example

Figure 10–15 shows an example of the use of proxy ARP. In this example, LAN A comprises an IP network in which subnetworks have not been implemented. The IP address of the network is 16.0.0.0.

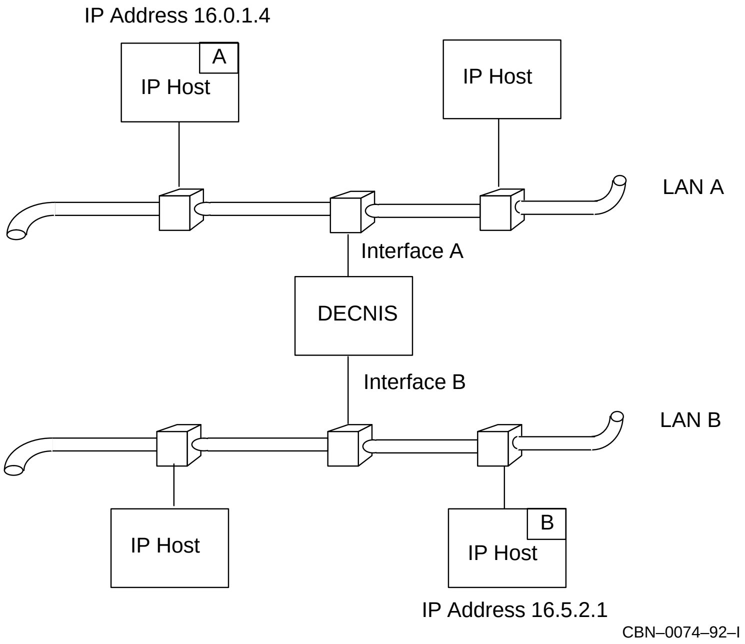
LAN B is an IP subnetwork, with an address of 16.5.0.0, and a subnet mask of 255.255.0.0.

IP Host A wishes to send an IP packet to IP Host B. Because IP Host A does not acknowledge subnetworks, it assumes that it can send an ARP request message to any IP host with a network address of 16.0.0.0, so it sends an ARP request message to ask for the hardware address of IP Host B.

If the DECNIS were not configured to use proxy ARP, IP Host A would not receive a response to its ARP request message.

However, with proxy ARP enabled on the Interfaces A and B, the DECNIS knows that it can reach IP Host B, and therefore responds to the ARP request message from IP Host A. IP Host A sends the IP packet destined for IP Host B to the DECNIS, and the DECNIS forwards it to IP Host B.

Figure 10–15 DECNIS Using Proxy ARP



10.23.4 Setting Up Proxy ARP on the DECNIS

To configure the DECNIS to use proxy ARP, issue the following command:

```
NCL> SET ROUTING CIRCUIT circuit-name PROXY ARP ON
```

In the example shown in Figure 10–15, you must enter the command on Interface A and Interface B.

10.24 Using Alternative Subnet Addresses

10.24.1 Introduction

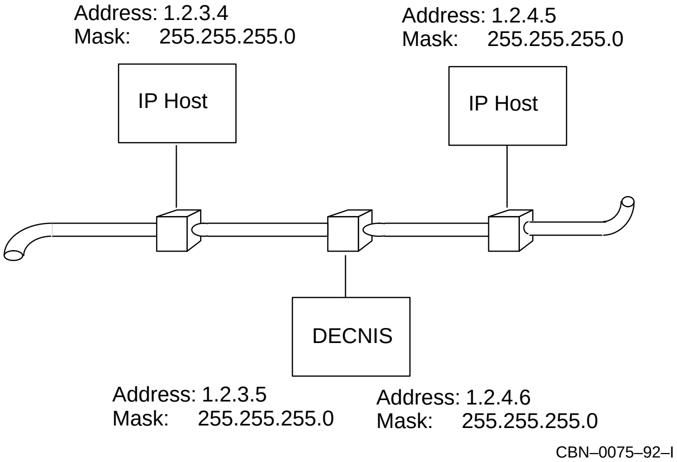
Normally, ARP messages are only sent between systems that are in the same subnet. If there are two or more subnets on the same LAN, the DECNIS can be configured to route messages between them, as shown in Figure 10–16.

To configure the DECNIS to route between different subnets on a LAN, you must give the DECNIS one or more alternative subnet addresses.

10.24.2 Example

In Figure 10–16, when IP Host 1.2.3.4 wants to communicate with IP Host 1.2.4.5, it sends a message to IP address 1.2.3.5.

Figure 10–16 Routing Between Subnets on a LAN



The DECNIS can forward the message to IP Host 1.2.4.5, from its IP address of 1.2.4.6.

10.24.3 Setting Up Alternative Subnet Addresses on the DECNIS

Set up alternative subnet addresses on the DECNIS as follows:

1. Disable the routing circuit:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name
```

2. Add alternative subnet addresses:

```
NCL> ADD ROUTING CIRCUIT circuit-name -
_NCL> ALTERNATIVE SUBNET ADDRESSES {[ADDRESS=(a.b.c.d), MASK=(e.f.g.h)]}
```

3. Enable the routing circuit:

```
NCL> ENABLE ROUTING CIRCUIT circuit-name
```

10.25 Configuring ICMP Router Discovery

The DECNIS implements ICMP Router Discovery, as defined in RFC 1256. ICMP Router Discovery enables IP hosts in a subnet to discover dynamically the addresses of routers in the subnet.

If you set up ICMP Router Discovery on DECNIS circuits, the DECNIS will periodically send out router advertisements on each circuit to hosts on connected subnet(s). The advertisements contain the IP address(es) for the circuit, together with a metric indicating how preferred each address is as a default router, relative to other router addresses on the subnet.

10.25.1 Requirements

Hosts on connected subnets must support ICMP Router Discovery.

10.25.2 Setting Up ICMP Router Discovery

To set up ICMP Router Discovery on the DECNIS, follow these steps:

1. Create the Router Discovery entity:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name ROUTER DISCOVERY
   ```

   where *circuit-name* is the name of an IP circuit.

2. Specify the IP addresses to be advertised on the circuit, with their preference levels:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name ROUTER DISCOVERY -
   _NCL> ADDRESSES {{ADDRESS=a.b.c.d,PREFERENCE=n}, -
   _NCL> {ADDRESS=e.f.g.h,PREFERENCE=m}}
   ```

   where: *a.b.c.d* and *e.f.g.h* are IP addresses assigned to this circuit.

   m and *n* are decimal numbers in the range 1 to 65,535, indicating the preference levels for *a.b.c.d* and *e.f.g.h* respectively. The higher the number, the more preferred the address.

3. Specify the destination address for router advertisements:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name ROUTER DISCOVERY -
   _NCL>  ADVERTISEMENT ADDRESS advert-address
   ```

   where *advert-address* is one of the following:

255.255.255.255	for IP limited broadcast
224.0.0.1	for IP multicast all hosts

   The default value is 224.0.0.1.

4. Enable the ROUTER DISCOVERY entity:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name ROUTER DISCOVERY
   ```

10.25.3 Defining Advertisement Intervals and Address Lifetime

The interval between router advertisements varies, in order to reduce the possibility of many routers sending advertisements at the same time. For this reason, you cannot specify the precise interval between advertisements. However, you can specify the maximum and minimum interval.

You can also define the lifetime of the addresses in router advertisements. The lifetime is the maximum length of time that advertised addresses are to be considered valid by hosts.

Procedure

To specify the maximum and minimum intervals between router advertisements, and the lifetime of advertised addresses, follow these steps:

1. Specify the maximum number of seconds the DECNIS is allowed to wait before sending a router advertisement:

   ```
   NCL>  SET ROUTING CIRCUIT circuit-name ROUTER DISCOVERY -
   _NCL>  MAX ADVERTISEMENT INTERVAL n
   ```

   where *n* is a decimal number in the range 4 to 1800. The default is 600 seconds.

2. Specify the minimum number of seconds the DECNIS must wait before sending a router advertisement:

   ```
   NCL>  SET ROUTING CIRCUIT circuit-name ROUTER  DISCOVERY -
   _NCL>  MIN ADVERTISEMENT INTERVAL n
   ```

   where *n* is a decimal number in the range from 3 to the value specified by MAX ADVERTISEMENT INTERVAL.

3. Specify the lifetime for advertised addresses:

   ```
   NCL>  SET ROUTING CIRCUIT circuit-name ROUTER  DISCOVERY -
   _NCL>  LIFETIME n
   ```

   where *n* is a decimal number of seconds in the range from the value specified by MAX ADVERTISEMENT INTERVAL to 9000.

Example

On circuit L602-4-0, you want to specify that the maximum interval between router advertisements on this circuit must be *no longer than* 60 seconds, and that the minimum interval between router advertisements on this circuit must be *at least* three seconds.

To do this, enter the following commands:

```
NCL> SET ROUTING CIRCUIT  L602-4-0 ROUTER DISCOVERY -
_NCL> MAX ADVERTISEMENT INTERVAL 60 -
_NCL>  MIN ADVERTISEMENT INTERVAL 3
```

10.26 Using SNMP to Manage IP Routing

The following sections describe some of the MIB variables in the Address Translation and IP groups of MIB-II, and their meaning in DECNIS terms.

atEntry: Adding to the ARP Cache

MIB: MIB-II, *MIB group:* Address Translation, *MIB table:* atTable

This table constitutes the ARP cache of the DECNIS. Entries that you add using SNMP remain in the ARP cache until manually removed or until the DECNIS is rebooted.

To create a new ARP cache entry for a particular interface, create a new entry, atEntry_*n.a.b.c.d*, where *n* is the value of the ifIndex for the interface, and *a.b.c.d* is the IP address to be translated. Each atEntry comprises the MIB variables atIfIndex, atNetAddress, and atPhysAddress (see below).

Note that you cannot create atEntry variables for X.25 interfaces.

atIfIndex

MIB: MIB-II, *MIB group:* Address Translation, *MIB table:* atTable

This is the index number of the DECNIS interface: it is the value of the ifIndex variable in the Interfaces group.

atNetAddress

MIB: MIB-II, *MIB group:* Address Translation, *MIB table:* atTable

This is the IP address of the remote system.

atPhysAddress

MIB: MIB-II, *MIB group:* Address Translation, *MIB table:* atTable

This is the physical address of the remote system, for example, AA-00-04-00-07-34. If the interface is X.25, the value displayed here will be the remote DTE address.

ipAddrEntry

MIB: MIB-II, *MIB group:* IP, *MIB table:* ipAddrTable

Alternative subnet addresses (see Section 10.24) will each have a separate entry. The value of ipAdEntIfIndex (the index number of the interface) will be the same for each.

ipRouteTable: IP Forwarding Table

MIB: MIB-II, *MIB group:* IP, *MIB table:* ipRouteTable

This table represents the IP forwarding table of the DECNIS. It specifies those IP routes that the DECNIS is using.

ipRouteEntry: Creating SNMP Routes

MIB: MIB-II, *MIB group:* IP, *MIB table:* ipRouteTable

Each entry in the forwarding table specifies a destination IP address. To place a new route in the forwarding table, create an ipRouteEntry_*a.b.c.d*, with an ipRouteDest of *a.b.c.d*, where *a.b.c.d* is the destination address. Specify the values for ipRouteIfIndex, ipRouteNextHop, and ipRouteMask (mandatory for broadcast circuits).

Note that if the forwarding table already contains a route to this address, this preexisting route will be dropped in favour of the one you insert.

You cannot create entries for X.25 dynamically assigned circuits.

Route Metrics

MIB: MIB-II, *MIB group:* IP, *MIB table:* ipRouteTable

ipRouteMetric2, ipRouteMetric3, ipRouteMetric4, and ipRouteMetric5 are not supported on the DECNIS.

udpLocalPort

MIB: MIB-II, *MIB group:* UDP, *MIB table:* udpTable

There will be a table entry with a udpLocalPort value of 161 when the DECNIS is using SNMP.

There will be an additional entry with a udpLocalPort value of 520 if the DECNIS is running RIP.

egpNeighEntry

MIB: MIB-II, *MIB group:* EGP, *MIB table:* egpNeighTable

Each entry corresponds to an EGP NEIGHBOR entity (see Section 10.13).

11

IP Route Propagation and Filtering

11.1 Introduction

This chapter describes how IP routes are received and propagated on the DECNIS, and how you can filter these routes. Route filtering controls how routes are received and advertised by the DECNIS.

Note that this chapter does not describe how to set up route filtering for the BGP protocol. Refer to Chapter 12 for a description of BGP route filtering.

11.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

Note

For any tasks that require you to disable a routing circuit, ensure that you do not disable the routing circuit that you are using to manage the DECNIS, unless there is an alternative route between the host on which you issue the NCL commands and the DECNIS.

11.2 Introduction to Receiving and Propagating Routes

The process for receiving and propagating IP routes is as follows:

- The DECNIS periodically receives route advertisements from its neighboring routers. By default, these are all entered into Routing Tables. There is a Routing Table for each routing protocol.

The best routes are passed on to the Forwarding Table, which is used for IP packet forwarding and route advertising. The best routes are calculated based on metric cost and protocol preference.

- IP routing protocols in the DECNIS periodically advertise routes from the Forwarding Table to neighboring routers. By default, only local routes and routes derived from the same routing protocol are advertised. In addition, link-state protocols such as IS–IS and OSPF always re-advertise any routes they receive.

The process described above may be influenced by:

- Setting up received route filtering

  By setting up received route filtering, you can decide which routes are accepted into the Routing Table and how they are accepted. Refer to Section 11.4.2 for further information.

- Setting up route propagation filtering

  By setting up route propagation filtering, you can decide which routes are advertised and override default settings. For example, you may want the RIP protocol instance to advertise a route that was learnt by OSPF. Refer to Section 11.6 for further information.

11.3 Entities Controlling Route Propagation and Filtering

Figure 11–1 shows the network management entities that control filtering of received and propagated routes. Table 11–1 explains each component in the figure.

Figure 11–1 Entities that Control Route Propagation and Filtering

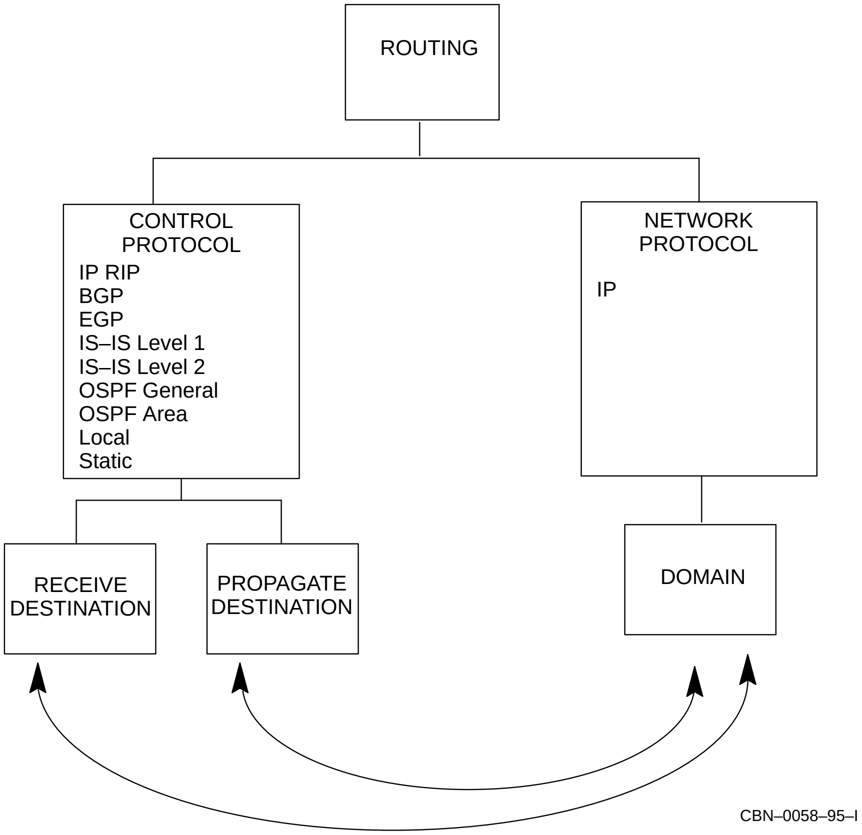


Table 11–1 Entity Descriptions

Entity	Description
CONTROL PROTOCOL	This is a subentity of the Routing module. A CONTROL PROTOCOL entity is associated with each IP routing protocol in the DECNIS. The TYPES of the entities are:

Route Type	Control Protocol TYPE	Created
BGP	BGP	When setting up BGP.
EGP	EGP	When setting up EGP route filtering or propagation.
IS–IS Level 1	IS-ISLEVEL1	Automatically when enabling Routing.
IS–IS Level 2 (and Level 2 routing is set up)	IS-ISLEVEL2	Automatically when enabling Routing.
OSPF Interarea	OSPF-Area	When setting up OSPF.
OSPF Intra-area	OSPF-Area	When setting up OSPF.
OSPF External Type 1	OSPF-General	When setting up OSPF.
OSPF External Type 2	OSPF-General	When setting up OSPF.
RIP	IP RIP	When setting up RIP.
Local	LOCAL	Automatically when enabling Routing.
Static	STATIC	Automatically when enabling Routing.

Entity	Description
RECEIVE DESTINATION	This is a subentity of the CONTROL PROTOCOL entity. It is set up to override the defaults for receiving route advertisements in the Routing Tables (and so influencing what routes go into the Forwarding Table).
PROPAGATE DESTINATION	This is a subentity of the CONTROL PROTOCOL entity. It is set up to override the defaults for advertising routes to neighboring routers.
NETWORK PROTOCOL	This is a subentity of the Routing module. In order to filter received and propagated routes, you must create a NETWORK PROTOCOL entity of type IP. This is created automatically when you set up IP using either DECNIS configurator.

(continued on next page)

Table 11–1 (Cont.) Entity Descriptions

Entity	Description
DOMAIN	This is a subentity of the NETWORK PROTOCOL entity. It is used to specify the addresses of IP routes to be filtered or propagated. The RECEIVE DESTINATION and PROPAGATE DESTINATION entities may refer to the DOMAIN entity to determine which routes to filter and propagate. If you want to filter specific received or propagated routes, you need to set up addressing information in a ROUTING NETWORK PROTOCOL IP DOMAIN entity. You can set up any number of IP domains. Each IP domain has an IP ADDRESS attribute which contains IP addresses and masks. An IP domain can contain one address and mask or a set of addresses and masks.

11.4 Received Route Filtering

11.4.1 What Happens Without Route Filtering?

If you do not set up any filtering for received routes, the following happens:

- Each route received from neighboring routers is placed in a Routing Table for its protocol.
- Each route is given a default preference according to its protocol.

Note

This does not apply to BGP. See Section 12.2.1.1 for details.

11.4.1.1 Advertisement Information Received

The information advertised by each neighboring router is listed in Table 11–2.

An example of route advertising is shown in Figure 11–2.

Figure 11–2 Route Advertising

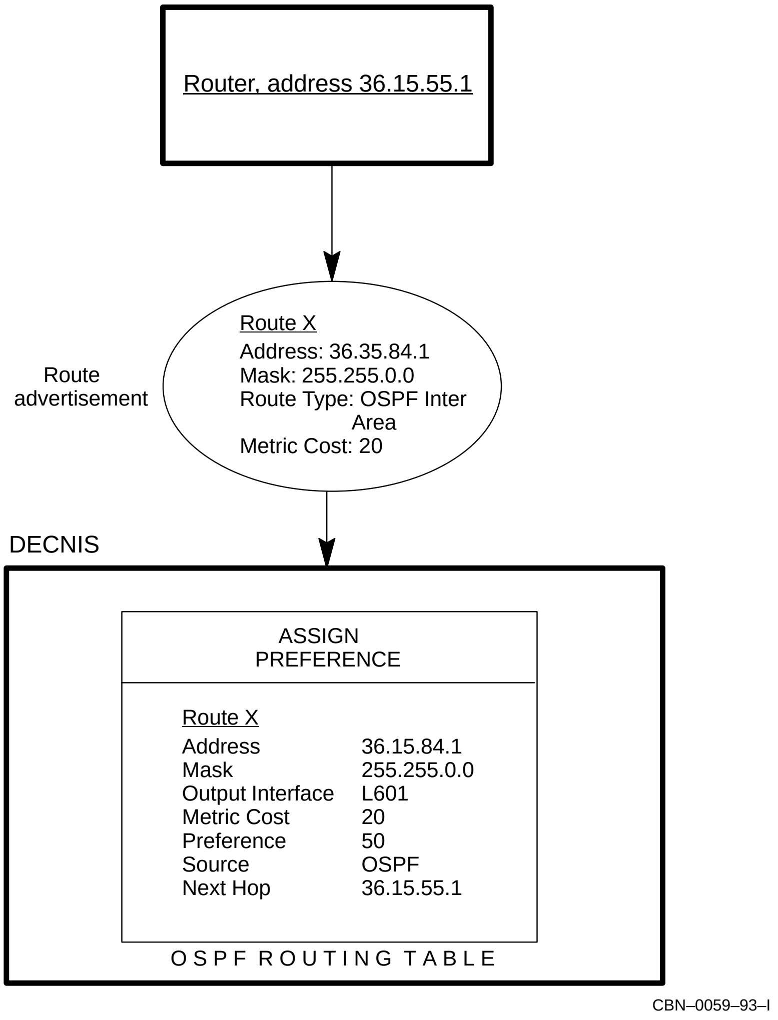


Table 11–2 Advertisement Information Received

<table>
<tr><th>Information</th><th>Explanation</th></tr>
<tr><td>Address</td><td>The IP address.</td></tr>
<tr><td>Mask</td><td>The IP address mask.</td></tr>
<tr><td>Route Type</td><td>The route type which may take the following values:
<table>
<tr><th>Routing Protocol</th><th>Route Types</th></tr>
<tr><td>BGP</td><td>BGP</td></tr>
<tr><td>EGP</td><td>EGP</td></tr>
<tr><td>IS–IS</td><td>IS–IS Level 1
IS–IS Level 2 Internal
IS–IS Level 2 External</td></tr>
<tr><td>OSPF</td><td>OSPF Intra-area</td></tr>
<tr><td>OSPF</td><td>OSPF Interarea</td></tr>
<tr><td>OSPF</td><td>OSPF External Type 1</td></tr>
<tr><td>OSPF</td><td>OSPF External Type 2</td></tr>
<tr><td>RIP</td><td>RIP</td></tr>
</table>
</td></tr>
<tr><td>Metric Cost</td><td>The cost of this route to the destination address.</td></tr>
</table>

Section 11.4.1.2 describes the information stored in the Routing Tables.

11.4.1.2 Routing Tables

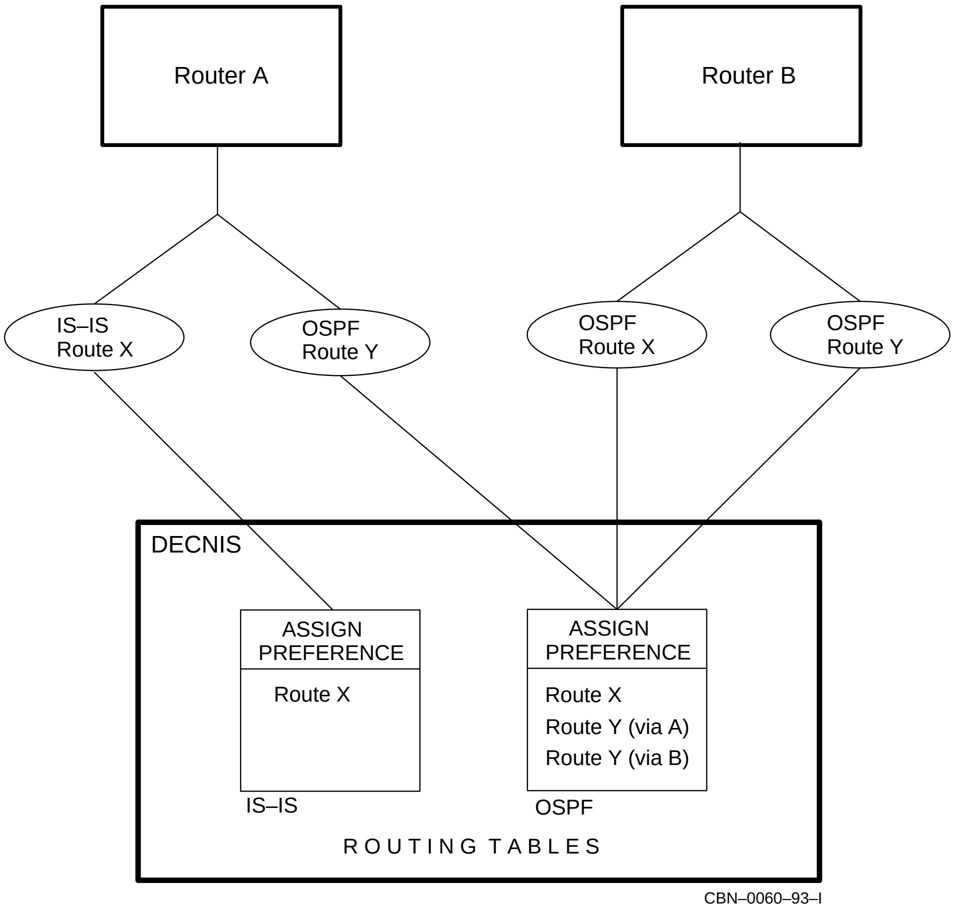
When the route advertisement arrives at the DECNIS, it is stored in the Routing Tables after a preference has been assigned. Each routing protocol has its own Routing Table. Routing Tables are invisible to the user. The information held in Routing Tables is listed in Table 11–3.

Table 11–3 Routing Table Data

<table>
<tr><th>Information</th><th>Explanation</th></tr>
<tr><td>Address</td><td>IP address as received from the route advertisement.</td></tr>
<tr><td>Mask</td><td>IP address mask as received from the route advertisement.</td></tr>
<tr><td>Output Interface</td><td>The DECNIS interface through which data should be sent out to this address. By definition this is the interface through which the route advertisement was received.</td></tr>
<tr><td>Metric Cost</td><td>The metric cost of the route as received from the route advertisement.</td></tr>
<tr><td>Preference</td><td>A number indicating the preference assigned to a particular route type. Each route received is assigned a preference. Preferences are of importance for the selection of the best routes which are passed on to the Forwarding Table. The lower the number, the more preferred the route.
Default route preferences are as follows:
<table>
<tr><th>Route Type</th><th>Default Preference</th></tr>
<tr><td>Local</td><td>0</td></tr>
<tr><td>Static</td><td>10</td></tr>
<tr><td>IS–IS Level 1</td><td>20</td></tr>
<tr><td>OSPF Intra-area</td><td>30</td></tr>
<tr><td>IS–IS Level 2 Internal</td><td>40</td></tr>
<tr><td>OSPF Inter Area</td><td>50</td></tr>
<tr><td>OSPF External Type 1</td><td>60</td></tr>
<tr><td>RIP</td><td>70</td></tr>
<tr><td>BGP</td><td>75</td></tr>
<tr><td>EGP</td><td>80</td></tr>
<tr><td>IS–IS Level 2 External</td><td>90</td></tr>
<tr><td>OSPF External Type 2</td><td>100</td></tr>
</table></td></tr>
<tr><td>Source</td><td>The Routing Protocol from which the advertisement was received.</td></tr>
<tr><td>Next Hop</td><td>The next address to which data should be forwarded after this address. This is usually the address of the router that sends the advertisement; however, in the case of OSPF a different, forwarding address may be given.</td></tr>
</table>

The RIP and EGP Routing Tables only contain one route to a particular destination, which is selected on the grounds of the lowest metric cost. The OSPF and IS–IS Routing Tables do not have such a selection process, and by default, any routes that are advertised are accepted. This means that a particular route may appear in a single Routing Table more than once (which means that it was advertised by more than one neighboring router), and it may appear in more than one Routing Table (which means that it was advertised for more than one routing protocol, through one or more routers). Figure 11–3 shows how this can happen.

Figure 11–3 Multiple Occurrences of a Route



In Figure 11–3, Route X appears in the IS–IS Routing Table through a route advertisement from Router A. Route X also appears in the OSPF Routing Table through a route advertisement from Router B.

Route Y appears in the OSPF Routing Table twice because it was advertised by both Router A and Router B.

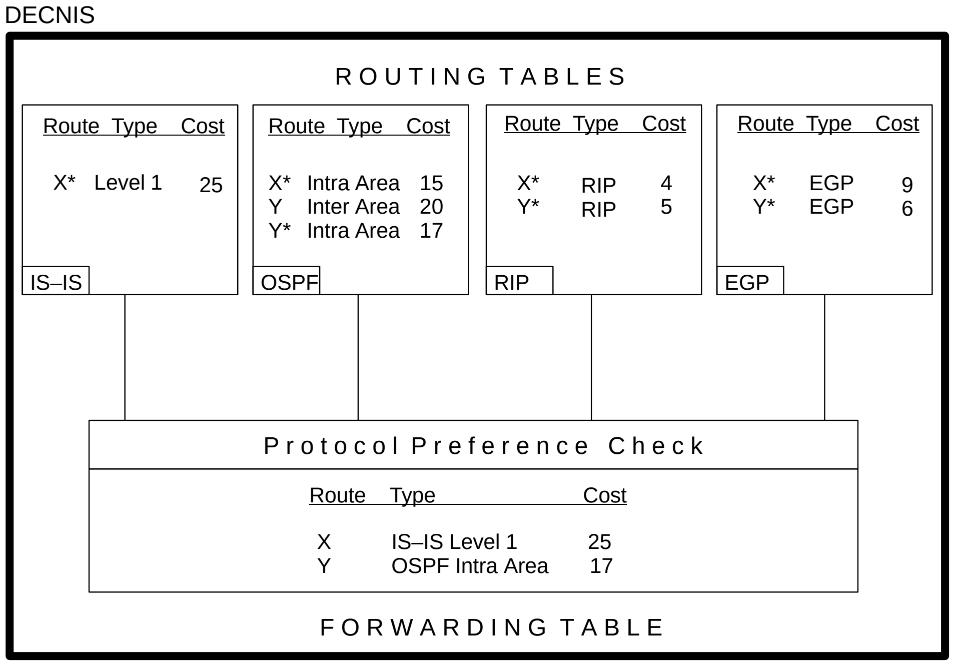
11.4.1.3 Routes Passed from the Routing Tables to the Forwarding Table

Periodically, the Routing Tables pass on the routes with the lowest metric cost to the Forwarding Table that is used for data forwarding and route propagation.

If the Forwarding Table receives a route more than once, that is, it receives the same route from more than one Routing Table, then it selects the route with the best preference. If they have equal preference, then another metric cost check is done and the route with the lowest metric cost selected for entry in the Routing Table. For each route, the Forwarding Table can contain up to four ‘equal cost routes’, that is, routes of the same metric cost and preference. Any other versions of the route are discarded.

An example of this process is shown in Figure 11–4. Routes of the lowest cost are marked with an asterisk (*). These are passed on to the Forwarding Table. The Forwarding Table receives four versions of Route X. In the protocol preference check, Route X of route type IS–IS Level 1 is found to be the most preferred route (best preference in Table 11–3), and is therefore inserted into the Forwarding Table.

Figure 11–4 Routes Passed to the Forwarding Table



CBN–0061–93–I

11.4.2 Filtering Received Routes

You can control routes to be inserted into the Forwarding Table by controlling which routes are inserted into the Routing Table, and how they are inserted. This is done by setting up a RECEIVE DESTINATION filter.

Route filtering is used to achieve the following:

- Alter the default preference of a route that is received.

  This controls which route the Forwarding Table will choose as the ‘most preferred’ route.

- Block certain routes

  Routes may be blocked to prevent them from being inserted into the Routing Table altogether.

11.5 Setting Up Filtering of Received Routes

This section describes how to configure the DECNIS to filter received routes. Note that BGP filtering of received routes is described in Chapter 12.

11.5.1 Tasks in Setting Up Route Filtering

To set up filtering of received routes, you must carry out the following tasks:

1. Create a NETWORK PROTOCOL entity of IP if this has not been done already. Enter the following command:

   ```
   NCL> CREATE ROUTING NETWORK PROTOCOL IP
   ```

2. Set up the list of addresses you want to block from entering the Routing Tables. To do this, you set up one or more IP domains.

   An IP domain is a set of IP address/mask pairs, specified in a DOMAIN subentity of the ROUTING NETWORK PROTOCOL IP entity. A single DOMAIN subentity can be reused by any number of receive or propagate filters.

   You can create any number of domains. Refer to Section 11.11.2 to find out how to set up an IP domain.

3. Set up a RECEIVE DESTINATION filter for the CONTROL PROTOCOL, using the IP domain(s) set up in step 2.

   Refer to Section 11.5.2 to find out how to set up a RECEIVE DESTINATION filter.

11.5.2 The RECEIVE DESTINATION Filter

Each RECEIVE DESTINATION has the characteristics shown in Table 11–4.

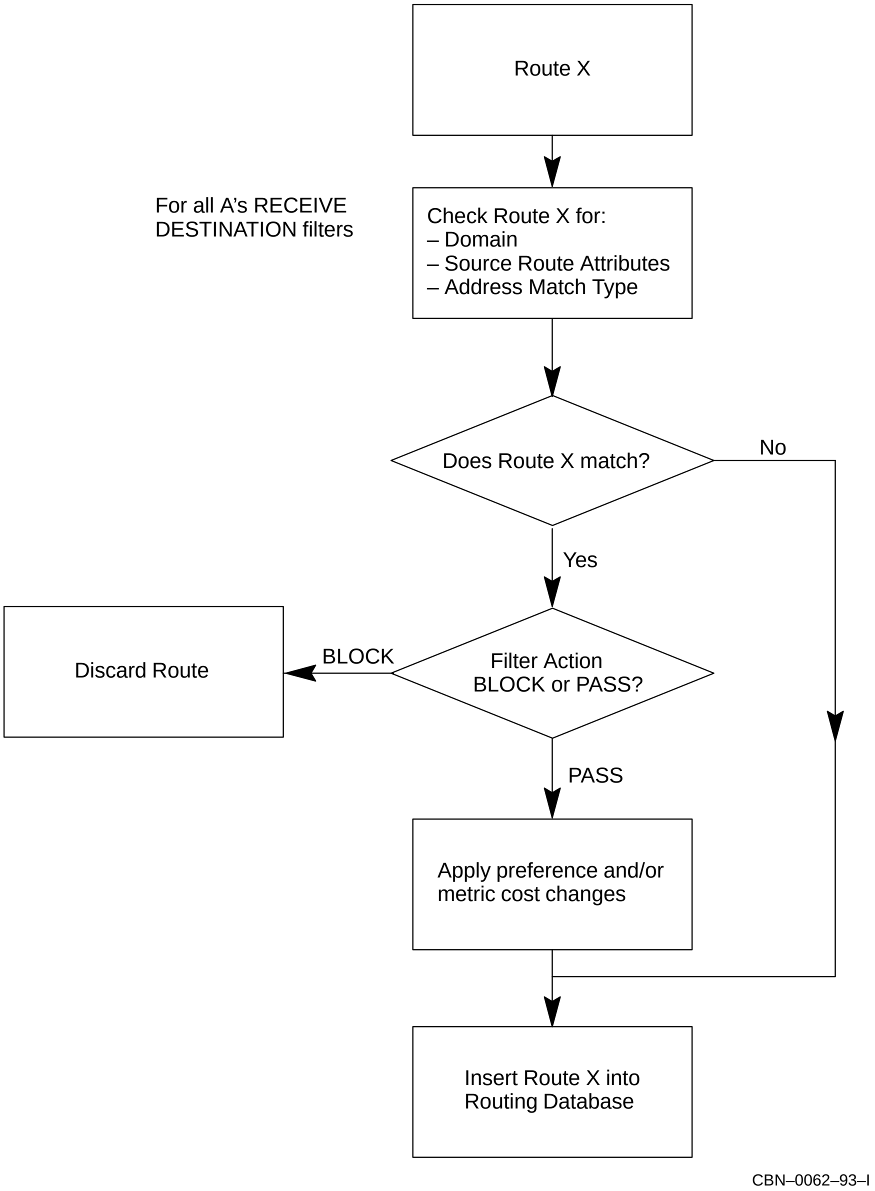
Table 11–4 RECEIVE DESTINATION Characteristics

For characteristic...	Specify...
DOMAINS	A list of DOMAIN entities containing the addresses and masks of the routes you want to filter.
ADDRESS MATCH TYPE	EXACT or PREFIX.
SOURCE ROUTE ATTRIBUTES	The type of route which you want to filter. Specify only for IS–IS Level 2 routes and all OSPF routes.
FILTER ACTION	BLOCK or PASS.
PREFERENCE	A new preference. Use in conjunction with a FILTER ACTION of PASS. You can change the default preference of any routing protocol. It is recommended that you do not give any two protocols the same preference value.

All characteristics except FILTER ACTION and PREFERENCE are used to check whether the incoming route advertisement matches the RECEIVE DESTINATION filter. If it does, then the FILTER ACTION, and/or any PREFERENCE changes are applied to the route.

Figure 11–5 shows the received route filtering process for CONTROL PROTOCOL ‘A’, which can be any IP control protocol instance.

Figure 11–5 Protocol Instance ‘A’ - Route Filtering


Route X
For all A’s RECEIVE DESTINATION filters
Check Route X for:
– Domain
– Source Route Attributes
– Address Match Type
Does Route X match?
No
Yes
Discard Route
BLOCK
Filter Action BLOCK or PASS?
PASS
Apply preference and/or metric cost changes
Insert Route X into Routing Database
CBN–0062–93–I

11.5.3 Setting Up a RECEIVE DESTINATION Filter to Block Routes

This section assumes that the following have been set up:

- A ROUTING NETWORK PROTOCOL IP entity. Refer to Section 11.5.1 for details.
- A ROUTING CONTROL PROTOCOL entity for each type of route you want to filter. Refer to Section 11.3 for details.

Follow these steps:

1. Create an IP domain specifying the routes that you want to block from being inserted into the Routing Table. Section 11.11.2 explains how to create an IP domain. If you do not specify a domain, all routes of that particular protocol will be blocked.
2. Create the RECEIVE DESTINATION filter by entering the following command:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> RECEIVE DESTINATION filter-name FILTER ACTION action, -
   _NCL> [SOURCE ROUTE ATTRIBUTES (s-r-attribute)]
   ```

   where:

   control-prot-name is the name of the control protocol instance.

   filter-name is the name of the RECEIVE DESTINATION filter.

   action is either BLOCK or PASS.

   s-r-attribute only applies when filtering the route types in Table 11–5.

   For example:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL IS-ISLEVEL2 -
   _NCL> RECEIVE DESTINATION filter_1 FILTER ACTION BLOCK, -
   _NCL> SOURCE ROUTE ATTRIBUTES (ISIS INTERNAL)
   ```

Table 11–5 Source Route Attributes

If the route type is...	Then the Source Route Attribute is...
IS–IS Level 2 External	ISIS External
IS–IS Level 2 Internal	ISIS Internal
OSPF Intra Area	OSPF Intra_Area
OSPF Inter Area	OSPF Inter_Area
OSPF External Type 1	OSPF External Type 1
OSPF External Type 2	OSPF External Type 2

3. Specify the IP domain(s) with the routes that you want to filter. You can specify one IP domain or a set of IP domains. Enter the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> RECEIVE DESTINATION filter-name DOMAINS {{ROUTING NETWORK -
   _NCL> PROTOCOL IP DOMAIN domain-name} ,...}
   ```

   For example:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL IS-ISLEVEL2 -
   _NCL> RECEIVE DESTINATION FILTER_1 DOMAINS {{ROUTING NETWORK -
   _NCL> PROTOCOL IP DOMAIN DOMAIN_1} , {ROUTING NETWORK PROTOCOL -
   _NCL> IP DOMAIN DOMAIN_2}}
   ```

   Refer to Section 11.11.2 to find out how to set up IP domains. For filtering all routes or the default route, do not specify any IP domains (see Section 11.5.7 and Section 11.5.8, respectively).

4. Specify the address match type:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> RECEIVE DESTINATION filter-name ADDRESS MATCH TYPE type
   ```

   where *type* is EXACT or PREFIX.

   For details about this attribute, see Section 11.11.4.

5. Now enable the RECEIVE DESTINATION filter by entering the following command:

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> RECEIVE DESTINATION filter-name
   ```

   Note

   You cannot set up OSPF or IS–IS RECEIVE DESTINATION entities on a running DECNIS. You must add the NCL commands to the appropriate user NCL script files and reboot the DECNIS. For details about editing the user NCL script files, refer to Section 1.5.6.

Refer to Section 11.9 for a example description of setting up filtering of received and propagated routes.

11.5.4 Setting Up a Filter to Change a Protocol Preference

You can set up a RECEIVE DESTINATION filter to change the protocol preference of some or all routes belonging to a CONTROL PROTOCOL. This affects which routes are inserted into the Forwarding Table (see Figure 11–4). Refer to Table 11–3 to find out what the default protocol preferences are.

For example, the default RIP protocol preference is 70. On your network, certain routes can be reached through RIP or IS–IS. Because of the default preferences, it is always the IS–IS version of the route that is inserted into the Forwarding Table. However, you may prefer the RIP version of the route, which is less congested.

To ensure that the RIP route is inserted into the Forwarding Table, you can change the protocol preference for RIP for that particular route, as follows:

1. Create an IP domain specifying the routes that will be inserted into the Routing Table at higher preference. Refer to Section 11.11.2 to find out how to create an IP domain. If you do not specify a domain, then all routes of that particular protocol will receive a higher preference.

2. Create a RECEIVE DESTINATION filter for the Control Protocol with FILTER ACTION PASS:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> RECEIVE DESTINATION filter-name FILTER ACTION PASS, -
   _NCL> [SOURCE ROUTE ATTRIBUTES (s-r-attribute)]
   ```

   where:

control-prot-name	is the name of the control protocol instance.
filter-name	is the name of the receive filter.
s-r-attribute	only applies when filtering the route types listed in Table 11–5.

   For our example:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL rip-1 -
   _NCL> RECEIVE DESTINATION elevate FILTER ACTION PASS
   ```

3. Specify the IP domain containing the routes that you want to receive at a higher preference:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> RECEIVE DESTINATION filter-name -
   _NCL> DOMAINS {ROUTING NETWORK PROTOCOL IP DOMAIN domain-name}
   ```

For our example:

```
NCL> SET ROUTING CONTROL PROTOCOL rip-1 -
_NCL> RECEIVE DESTINATION elevate -
_NCL> DOMAINS {ROUTING NETWORK PROTOCOL IP DOMAIN domain-1}
```

4. Set the preference:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> RECEIVE DESTINATION filter-name PREFERENCE n
   ```

   where *n* is the value of the new preference. For example:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL rip-1 -
   _NCL> RECEIVE DESTINATION elevate PREFERENCE 15
   ```

5. Enable the RECEIVE DESTINATION filter:

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> RECEIVE DESTINATION filter-name
   ```

   For example:

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL rip-1 -
   _NCL> RECEIVE DESTINATION elevate
   ```

   The preference of the RECEIVE DESTINATION will now be applied to the relevant routes.

Time Delays After Entering New Preference Values

There will be a short delay after re-enabling a CONTROL PROTOCOL and its associated filters before routes with the new preference are entered from the Routing Table into the DECNIS Forwarding Table.

Each routing protocol type has a slightly different delay time:

RIP	Within 30 seconds
EGP	Within 120 seconds

This time delay is not applicable to IS–IS, OSPF and BGP. The preference of IS–IS and OSPF routes can only be changed if Routing is disabled.

11.5.5 Disappearing Routes

The preference system can seem to make routes disappear from the DECNIS Forwarding Table.

For example, if the DECNIS first receives a RIP route with the address 16.36.16.0, it inserts it in the Forwarding Table. If the DECNIS later receives an IS–IS Level 2 Internal route, also with the address 16.36.16.0, it will delete the RIP route from the Forwarding Table and replace it with the IS–IS Level 2 Internal route (assuming default preferences).

This means that the DECNIS no longer uses the RIP instance of the route for forwarding. It *appears* that the RIP route has disappeared. This is expected behavior and since the DECNIS still has a route to the destination, there is no effect on forwarding of IP packets.

However, this does affect route propagation. For example, if the RIP route was being propagated into EGP, the propagation cannot now occur because the RIP route is no longer in the Forwarding Table.

11.5.5.1 Checking the Forwarding Table

To check the routes in the Forwarding Table, enter the following command:

```
NCL> SHOW ROUTING IP DESTINATION ADDRESS * ALL
```

11.5.6 Special Note About Filtering IS–IS and OSPF Routes

You should take extra care when setting up IS–IS and OSPF filtering, as it is possible to create "black holes" in your IS–IS network. This is because in link state protocols such as IS–IS and OSPF, the DECNIS must pass on routes regardless of whether it has blocked the route (the route does not exist in its Routing Table). Figure 11–6 shows an example of how a "black hole" can be created when using IS–IS filtering.

In Figure 11–6, system A tries to reach system X through DECNIS Y because this is the least cost path. However, DECNIS Y has been set up to block route X which, as a consequence, is not in its Forwarding Table. However, DECNIS Y *does* readvertise any IS–IS and OSPF routes it receives (irrespective of whether the route was blocked or not), including route X.

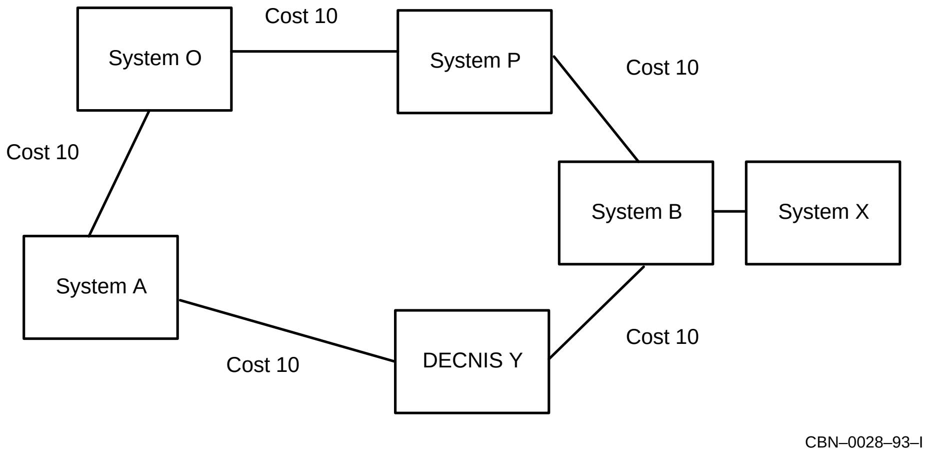
When system A tries to use DECNIS Y to reach system X, it cannot, because DECNIS Y does not have route X in its Forwarding Table.

11.5.7 Filtering All Routes of a Control Protocol

If you want all the routes of a particular control protocol type to match a filter:

- Do not specify the name of an IP domain.
- Set the ADDRESS MATCH TYPE characteristic to PREFIX.

Figure 11–6 IS–IS/OSPF Filtering



11.5.8 Filtering the Default Route

The default route has an address of 0.0.0.0 and a mask of 0.0.0.0. By default, the default route matches a filter when the following situation exists:

- No IP domain is specified.
- The ADDRESS MATCH TYPE characteristic is set to EXACT.

11.6 Filtering Routes for Propagation

11.6.1 What Happens Without Filtering of Propagated Routes?

If you do not filter propagated routes, then every few seconds (depending on the protocol type), the IP routing protocols in the DECNIS advertise the routes in the Forwarding Table to neighboring routers. Each IP protocol instance examines every route in the Forwarding Table to see if it may be advertised.

By default, an IP routing protocol will only advertise the following routes:

- Local Routes

  These are routes to destinations which are physically attached to the DECNIS.

- Routes derived from the same routing protocol instance

These are routes learnt by the same protocol instance, for example, EGP, IS–IS Level 1, OSPF General.

11.6.2 Purpose of Route Propagation Filtering

The purpose of filtering propagated routes for an IP protocol instance is:

- To change the metric used when propagating routes.
- To advertise routes learnt by another IP routing protocol. For example, you can set up route propagation for a RIP instance so that it advertises certain routes which were learnt by IS–IS.

Note that any route that you want to be propagated must be in the Forwarding Table of the DECNIS.

11.6.3 Advertising Routes Learnt by Another Routing Protocol

In the example in Figure 11–7, DECNIS A is receiving RIP routes from an IP router on LAN A. DECNIS A is also exchanging IP routing information with DECNIS B using the IS–IS protocol. You can set up route propagation for the relevant IS–IS protocol instance on DECNIS A so that it advertises the RIP routes (the routes that were learnt by the RIP protocol instance). In this way, DECNIS B learns about the systems on LAN A. This is referred to as propagating RIP routes into IS–IS.

See Section 11.9 for example NCL commands to set up RIP propagation.

For details about RIP instances and EGP instances, refer to Section 10.10 and Section 10.13.4.

11.7 Setting Up Route Propagation

This section describes how to configure the DECNIS to for route propagation. Note that BGP route propagation is described in Chapter 12.

11.7.1 Tasks in Setting Up Route Propagation

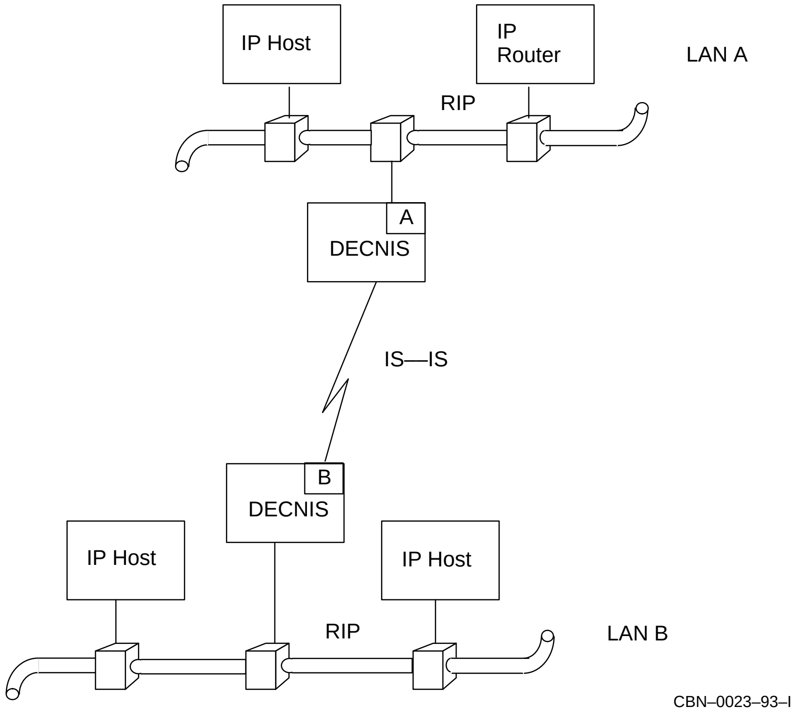
To set up route propagation, you must carry out the following tasks:

1. Create a NETWORK PROTOCOL entity of IP, if this has not been done already. Enter the following command:

   ```
   NCL> CREATE ROUTING NETWORK PROTOCOL IP
   ```

2. Set up the list of addresses you want to propagate. To do this, you set up one or more IP domains.

Figure 11–7 Route Propagation



An IP domain is a set of IP address/mask pairs, specified in a DOMAIN subentity of the ROUTING NETWORK PROTOCOL IP entity. A single DOMAIN subentity can be reused by more than one RECEIVE DESTINATION and/or PROPAGATE DESTINATION filter.

You can create any number of domains. Refer to Section 11.11.2 to find out how to set up an IP domain.

3. Set up a PROPAGATE DESTINATION filter for the CONTROL PROTOCOL, using the IP domain(s) set up in step 2.

   Refer to Section 11.7.2 to find out how to set up a PROPAGATE DESTINATION filter.

Note

In order to set up a PROPAGATE DESTINATION filter for OSPF, you must set up the DECNIS as an Autonomous System Boundary Router.

11.7.2 The PROPAGATE DESTINATION Filter

Each PROPAGATE DESTINATION filter has the characteristics listed in Table 11–6:

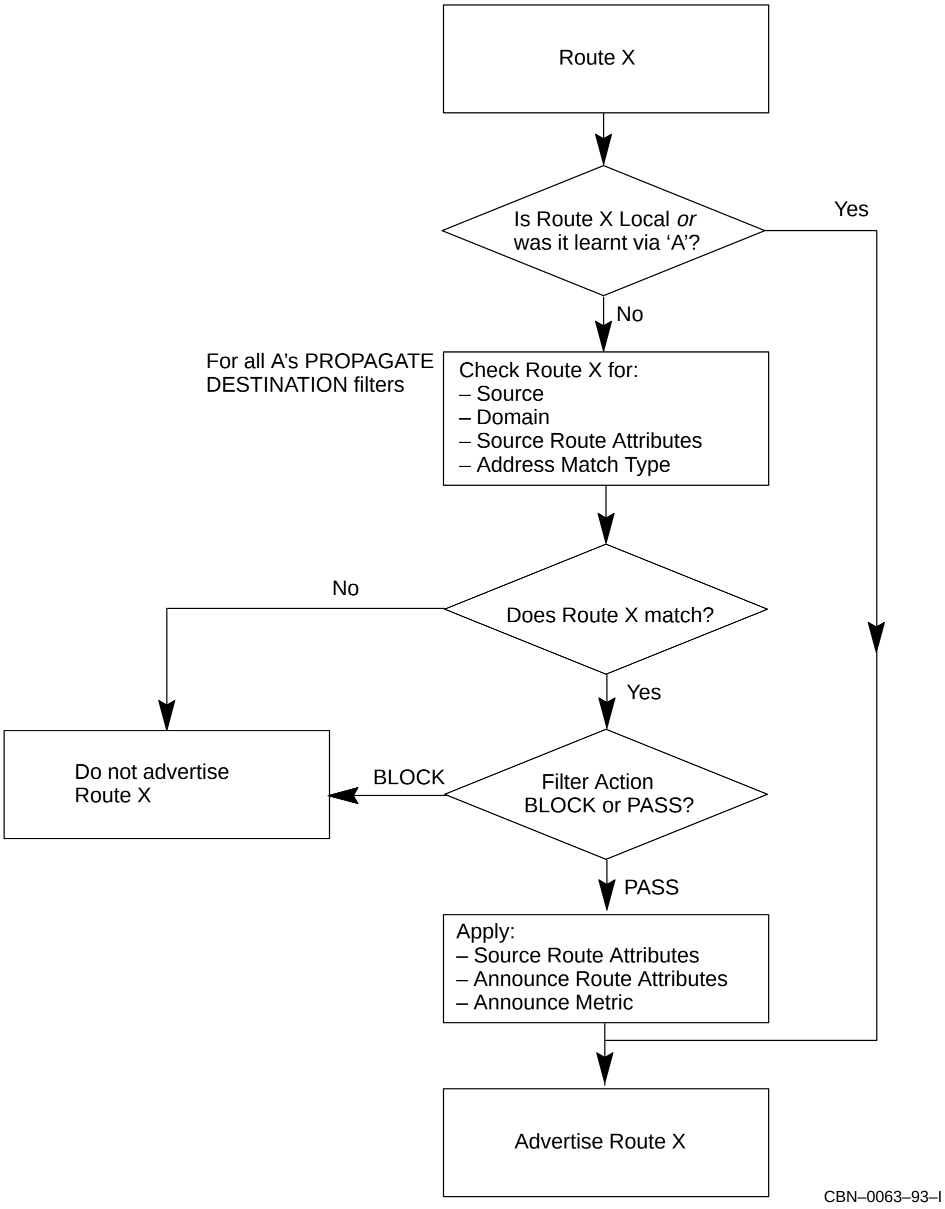
Table 11–6 PROPAGATE DESTINATION Characteristics

For characteristic...	Specify...
DOMAINS	Any of the IP domains that you have created containing the addresses and masks of the routes you want to filter.
SOURCES	The names of the CONTROL PROTOCOL entities whose routes you want to filter. In order to match, the route in the Forwarding Table must come from the source specified by SOURCES.
SOURCE ROUTE ATTRIBUTES	Specify only for IS–IS Level 2 routes and all OSPF routes. The type of route (for the control protocol(s) specified in SOURCES) you want to filter.
ADDRESS MATCH TYPE	EXACT or PREFIX.
FILTER ACTION	BLOCK or PASS.
ANNOUNCE METRIC	The metric with which the route will be advertised.
ANNOUNCE ROUTE ATTRIBUTES	The new route type with which the route will be advertised. Specify only for PROPAGATE DESTINATION entities belonging to IS–IS Level 2 and OSPF protocol instances.

The first four characteristics are used to check whether the route matches the PROPAGATE DESTINATION filter. If it does, it will be treated in the way specified by the FILTER ACTION characteristic. If the FILTER ACTION is PASS, then the route will be advertised with the metric specified in ANNOUNCE METRIC and, where applicable, the ANNOUNCE ROUTE ATTRIBUTES specified.

Figure 11–8 shows route propagation within the route advertising process for CONTROL PROTOCOL ‘A’ which can be any of the IP Routing CONTROL PROTOCOLs.

Figure 11–8 Protocol Instance ‘A’—Route Advertising


Route X
Is Route X Local *or* was it learnt via ‘A’?
Yes
No
For all A’s PROPAGATE DESTINATION filters
Check Route X for:
– Source
– Domain
– Source Route Attributes
– Address Match Type
No
Does Route X match?
Yes
Do not advertise Route X
BLOCK
Filter Action BLOCK or PASS?
PASS
Apply:
– Source Route Attributes
– Announce Route Attributes
– Announce Metric
Advertise Route X
CBN–0063–93–I

11.7.3 Setting Up a PROPAGATE DESTINATION Filter

This section assumes that the following have been set up:

- A ROUTING NETWORK PROTOCOL IP entity. Refer to Section 11.5.1.
- A ROUTING CONTROL PROTOCOL entity for each type of route you want to propagate or prevent from being propagated. Refer to Section 11.3.

To set up route propagation on the DECNIS, follow these steps:

1. Create an IP domain specifying the routes that you want to propagate. Refer to Section 11.11 to find out how to set up an IP domain.

   Do not set up an IP domain if you want to propagate all routes or the default route.

2. Create a PROPAGATE DESTINATION filter, specifying the filter action and source route attributes:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> PROPAGATE DESTINATION filter-name FILTER ACTION action, -
   _NCL> [SOURCE ROUTE ATTRIBUTES (s-r-attribute)]
   ```

   where:

   control-prot-name is the name of the protocol instance for which you are setting up this PROPAGATE DESTINATION. Note the following:

   - For the following TYPEs of CONTROL PROTOCOL instance, the name is the same as the TYPE:

     IS-ISLEVEL1
     IS-ISLEVEL2
     LOCAL
     STATIC

   - You cannot set up a PROPAGATE DESTINATION filter for an OSPF Area CONTROL PROTOCOL.

   filter-name is the name of the PROPAGATE DESTINATION filter.

   action is either BLOCK or PASS.

   s-r-attribute only applies when propagating IS–IS Level 2 and OSPF routes. The ISIS attributes can be either ISIS INTERNAL or ISIS EXTERNAL. The OSPF attributes can be either OSPF External Type 1 or OSPF External Type 2.

   For example, the following command creates a PROPAGATE DESTINATION for a RIP protocol instance with the name **rip_1** to advertise ISIS INTERNAL routes:

```
NCL> CREATE ROUTING CONTROL PROTOCOL rip_1 -
_NCL> PROPAGATE DESTINATION filter_1 FILTER ACTION PASS, -
_NCL> SOURCE ROUTE ATTRIBUTES (ISIS INTERNAL)
```

3. Specify the IP domain containing the routes you want to propagate. You can specify one IP domain or a set of IP domains.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> PROPAGATE DESTINATION filter-name DOMAINS {{ROUTING NETWORK -
   _NCL> PROTOCOL IP DOMAIN domain-name} ,...}
   ```

   For example:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL is-islevel2 -
   _NCL> PROPAGATE DESTINATION filter_1 DOMAINS {{ROUTING NETWORK -
   _NCL> PROTOCOL IP DOMAIN domain_1} , {ROUTING NETWORK PROTOCOL -
   _NCL> IP DOMAIN domain_2}}
   ```

   For propagating all routes or the default route, do not specify any IP domains. For more details, see Section 11.11.4 and Section 11.7.4.

4. Specify whether the address match type is to be EXACT or PREFIX:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> PROPAGATE DESTINATION filter-name ADDRESS MATCH TYPE type
   ```

   where *type* is EXACT or PREFIX.

   For details about this characteristic, see Section 11.11.4.

5. Specify the SOURCES characteristic.

   The SOURCES characteristic specifies the names of the CONTROL PROTOCOL instances that you want to propagate from. That is, it specifies the protocol instances which learnt the routes that you now want to advertise for this protocol. Enter the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> PROPAGATE DESTINATION filter-name SOURCES {{ROUTING CONTROL -
   _NCL> PROTOCOL control-prot-name} ,...}
   ```

   For example, you have created ROUTING CONTROL PROTOCOL entities for RIP and EGP with the names **Rip_routes** and **Egp_routes**. To propagate RIP and EGP routes to IS–IS Level 2, enter the following:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL IS-ISLEVEL2 -
   _NCL> PROPAGATE DESTINATION filter_1 SOURCES {{ROUTING CONTROL -
   _NCL> PROTOCOL Rip_routes}, {ROUTING CONTROL PROTOCOL Egp_routes}}
   ```

6. Specify the ANNOUNCE ROUTE ATTRIBUTES characteristic if this PROPAGATE DESTINATION filter is for an IS–IS or OSPF protocol instance. In this case, you need to specify what type of route the propagated route needs to be advertised as.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> PROPAGATE DESTINATION filter-name ANNOUNCE ROUTE -
   _NCL> ATTRIBUTES (a-r-attribute)
   ```

   where you may set *a-r-attribute* to the values in Table 11–7.

Table 11–7 ANNOUNCE ROUTE Attributes

To advertise the route as...	Then set the ANNOUNCE ROUTE ATTRIBUTE to...
IS–IS Level 2 External	ISIS External
IS–IS Level 2 Internal	ISIS Internal
OSPF External Type 1	OSPF External Type 1
OSPF External Type 2	OSPF External Type 2

   The default ANNOUNCE ROUTE attribute for an IS–IS Level 2 PROPAGATE DESTINATION filter is ISIS External. The default ANNOUNCE ROUTE attribute for an OSPF PROPAGATE DESTINATION filter is OSPF External Type 2.

7. Specify the cost that will be advertised with the route by setting the ANNOUNCE METRIC value. Enter the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> PROPAGATE DESTINATION filter-name ANNOUNCE METRIC value
   ```

   You should use caution when setting the ANNOUNCE METRIC value. Make sure that you do not set it to a value greater than the maximum for the protocol you are propagating into. If you specify a value greater than the maximum, the DECNIS will ignore this value and use a value equal to or less than the maximum. Table 11–8 specifies the maximum metric values for each type of protocol.

   If you set the ANNOUNCE METRIC value to 0, the DECNIS will use the metric that is set in the Forwarding Table. To check the metric for the routes in the Forwarding Table, enter the following command:

   ```
   NCL> SHOW ROUTING IP DESTINATION ADDRESS * ALL
   ```

   If you do not specify an ANNOUNCE METRIC value, then the DECNIS uses the default metric as specified in Table 11–8.

8. Now enable the PROPAGATE DESTINATION filter by entering the following command:

```
NCL> ENABLE ROUTING CONTROL PROTOCOL control-prot-name -
_NCL> PROPAGATE DESTINATION filter-name
```

Table 11–8 ANNOUNCE METRIC Values

If propagating into...	The maximum metric is...	And the default ANNOUNCE METRIC value is...
IS–IS Level 1	63	20
IS–IS Level 2	63	20
OSPF-General	16777214	20
RIP	15	1
EGP	255	1

11.7.4 Advertising a Default Route

The DECNIS can be set up to advertise a default route using route propagation. Follow the procedures in Section 11.7.3 using the following settings:

- Do not specify an IP domain
- Set the type of address matching to EXACT
- Set the SOURCES characteristic to LOCAL

11.7.5 Advertising All Routes of a SOURCE Control Protocol

A PROPAGATE DESTINATION filter can be set up in such a way that it advertises all the routes learnt by another control protocol instance. Follow the procedures in Section 11.7.3 using the following settings:

- Do not specify an IP domain
- Set the type of address matching to PREFIX

11.7.6 IS–IS Propagation Exceptions

There are two IS–IS propagation exceptions:

- You cannot propagate from EGP to IS–IS Level 1. In other words, you cannot set up a PROPAGATE DESTINATION filter for IS–IS Level 1 to advertise routes learnt by EGP.

- If Level 1 and Level 2 routing are set up on the DECNIS, you cannot propagate into IS–IS Level 1. In other words, if Level 1 and Level 2 routing are set up on the DECNIS, you cannot set up a PROPAGATE DESTINATION filter for IS–IS Level 1.

11.7.7 OSPF Propagation Exceptions

You cannot propagate into an OSPF AREA CONTROL PROTOCOL. You can only propagate routes into the OSPF GENERAL CONTROL PROTOCOL (OSPF-General).

11.8 Using the DECNIS Text-based Configurator to Set Up Route Propagation

The DECNIS text-based configurator allows you to set up a simple form of route propagation for all protocols except OSPF and BGP. This section describes how to use the DECNIS text-based configurator text-based configurator to do this.

The clearVISN DECNIS configurator allows you to set up full route propagation and received route filtering for Integrated IS–IS, RIP and OSPF. Refer to *clearVISN DECNIS Configurator User Guide* and the clearVISN DECNIS configurator on line help for details.

11.8.1 How the DECNIS Text-Based Configurator Sets Up Route Propagation and Filtering

The configurator asks which protocols you want to propagate. It then sets up the appropriate NCL commands to propagate the default route of the protocols that you select. If you specify propagation from IS–IS Level 2 routes, it will set the Source Route attributes to ISIS Internal.

If you require a more complicated form of propagation, you could use the configurator to set up the basic NCL commands and then add to them by editing the user NCL script files. For details about the user NCL script files, refer to Section 1.5.6.

There are example NCL scripts for setting up route propagation provided with the DECNIS software. Appendix G lists the directories holding these scripts.

If you select propagation into RIP, the configurator sets up a propagate destination for each RIP instance that exists. For example, if you used the configurator to set up a RIP Receive only circuit and a RIP Send and Receive circuit, there will be two RIP instances: RIPRx and RIPTxRx.

11.8.2 Example NCL Commands Created by the DECNIS Text-Based Configurator

The example below shows the NCL commands that will be created when propagation is specified:

- From RIP to IS–IS Level 2 (the RIP protocol is a receive only instance)
- From IS–IS Level 2 to EGP (the EGP protocol is associated with Autonomous System Number 111)

```
! Define all the required address domains that will be used
!
create routing network protocol IP
create routing network protocol IP domain DEFAULT_ROUTE_DOMAIN
set routing network protocol IP domain DEFAULT_ROUTE_DOMAIN -
    ip address {{addr= 0.0.0.0 ,mask= 0.0.0.0 } }
!
! Create PROPAGATE DESTINATIONS to define propagation of routes between
! protocols
!
! create PROPAGATE DESTINATION for routes from  RipRx  to IS-ISlevel2
! that will PASS when PREFIX match on domains DEFAULT_ROUTE_DOMAIN
!
create routing control protocol IS-ISlevel2 -
    propagate destination RipRx_TO_IS-ISlevel2_PASS_PREFIX_1 -
    filter action PASS
set routing control protocol IS-ISlevel2 -
    propagate destination RipRx_TO_IS-ISlevel2_PASS_PREFIX_1 -
    address match type PREFIX ,sources {{routing control protocol RipRx -
    } },domain {{routing network protocol IP domain DEFAULT_ROUTE_DOMAIN } }
!
! create PROPAGATE DESTINATION for routes from  IS-ISlevel2  to EGP111
! that will PASS when PREFIX match on domains DEFAULT_ROUTE_DOMAIN
!
create routing control protocol EGP111 -
    propagate destination IS-ISlevel2_TO_EGP111_PASS_PREFIX_5 -
    filter action PASS ,source route attributes { ISIS Internal }
set routing control protocol EGP111 -
    propagate destination IS-ISlevel2_TO_EGP111_PASS_PREFIX_5 -
    address match type PREFIX -
    ,sources {{routing control protocol IS-ISlevel2 -
    } },domain {{routing network protocol IP domain DEFAULT_ROUTE_DOMAIN } }
```

To determine how route propagation has been set up by the configurator, look at the master NCL script that is created when you exit the configurator.

If you want to change any of the route propagation settings, you must edit the user NCL script files and reboot the DECNIS. For details about the user NCL script files, refer to Section 1.5.6.

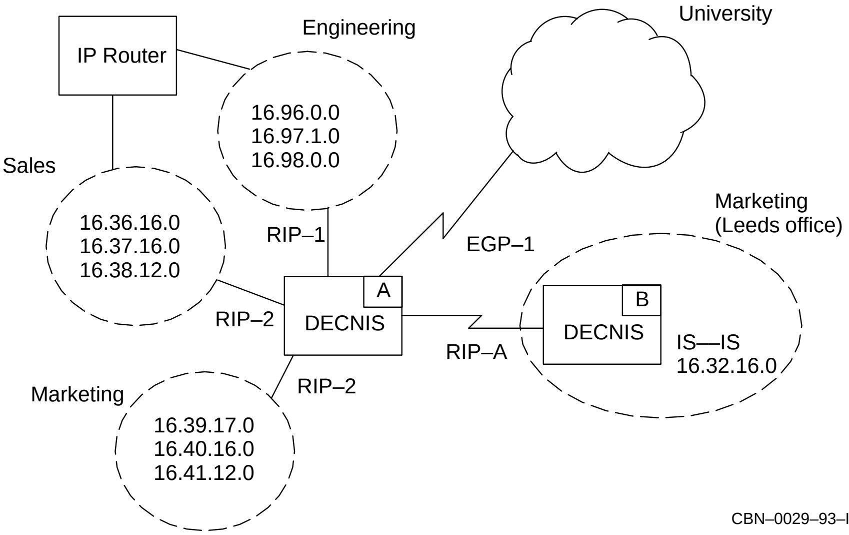
11.9 Examples of Setting Up Route Filtering Using RIP, EGP, and IS–IS

Figure 11–9 shows an example IP network that might exist within a company. The following examples describe how you might filter received and propagated routes to control the flow of IP routing traffic through the network.

Note

You cannot set up route filtering for Local routes.

Figure 11–9 Example IP Network Filtering Received and Propagated Routes



All the examples assume that the following entities have been created and enabled:

- NETWORK PROTOCOL IP
- ROUTING CONTROL PROTOCOL for the control protocol instance

Example 1

In Figure 11–9, the Engineering and Sales departments consist of several IP subnets. rip-1 is associated with subnet 16.96.0.0 in the Engineering department and RIP-2 is associated with subnet 16.36.16.0 in the Sales department.

The DECNIS receives route 16.36.16.0 from the Sales department, through RIP-2. It also receives route 16.36.16.0 from the Engineering department through the IP router and then rip-1. If you wanted the DECNIS to ignore advertisements of route 16.36.16.0 from the Engineering department, you could set up a RECEIVE DESTINATION filter to block the route.

The receive filter should be set up on DECNIS A to block any rip-1 routes with the prefix 16.36.16.0.

To set up this filter, enter the following commands:

- Create a domain containing the address and mask:

```
NCL> CREATE ROUTING NETWORK PROTOCOL IP DOMAIN RIP_ENG

NCL> SET ROUTING NETWORK PROTOCOL IP DOMAIN RIP_ENG -
_NCL> IP ADDRESS {{ADDRESS=16.36.16.0, MASK=255.255.255.0}}
```

- Create a receive filter:

```
NCL> CREATE ROUTING CONTROL PROTOCOL rip-1 -
_NCL> RECEIVE DESTINATION FILTER_RIP_ENG FILTER ACTION BLOCK

NCL> SET ROUTING CONTROL PROTOCOL rip-1 RECEIVE DESTINATION -
_NCL> FILTER_RIP_ENG ADDRESS MATCH TYPE PREFIX

NCL> SET ROUTING CONTROL PROTOCOL rip-1 -
_NCL> RECEIVE DESTINATION FILTER_RIP_ENG DOMAIN {ROUTING -
_NCL> NETWORK PROTOCOL IP DOMAIN RIP_ENG}

NCL> ENABLE ROUTING CONTROL PROTOCOL rip-1 -
_NCL> RECEIVE DESTINATION FILTER_RIP_ENG
```

Example 2

If there was only one route that you wanted to receive from the university, you could set up a filter to pass only that route. If the route you want to receive is 192.4.7.0, enter the following commands to set up an appropriate filter:

- Create an IP domain containing the address and mask of the route you want to receive:

```
NCL> CREATE ROUTING NETWORK PROTOCOL IP DOMAIN EGP_PASS

NCL> SET ROUTING NETWORK PROTOCOL IP DOMAIN EGP_PASS -
_NCL> IP ADDRESS {{ADDRESS=192.4.7.0, MASK=255.255.255.0}}
```

- Create a receive filter to block all EGP routes:

```
NCL> CREATE ROUTING CONTROL PROTOCOL EGP-1 -
_NCL> RECEIVE DESTINATION FILTER_EGP_BLOCK FILTER ACTION BLOCK

NCL> SET ROUTING CONTROL PROTOCOL EGP-1 -
_NCL> RECEIVE DESTINATION FILTER_EGP_BLOCK ADDRESS MATCH TYPE PREFIX

NCL> ENABLE ROUTING CONTROL PROTOCOL EGP-1 -
_NCL> RECEIVE DESTINATION FILTER_EGP_BLOCK
```

- Create another receive filter to pass the EGP route you want to receive:

```
NCL> CREATE ROUTING CONTROL PROTOCOL EGP-1 -
_NCL> RECEIVE DESTINATION FILTER_EGP_PASS FILTER ACTION PASS

NCL> SET ROUTING CONTROL PROTOCOL EGP-1 -
_NCL> RECEIVE DESTINATION FILTER_EGP_PASS DOMAIN {ROUTING -
_NCL> NETWORK PROTOCOL IP DOMAIN EGP_PASS}

NCL> SET ROUTING CONTROL PROTOCOL EGP-1 -
_NCL> RECEIVE DESTINATION FILTER_EGP_PASS ADDRESS MATCH TYPE EXACT

NCL> ENABLE ROUTING CONTROL PROTOCOL EGP-1 -
_NCL> RECEIVE DESTINATION FILTER_EGP_PASS
```

Example 3

If you wanted the Engineering department to be able to reach the subnet 192.4.7.0 within the university, you could set up a PROPAGATE DESTINATION filter by following these steps:

- You have already created the domain EGP_PASS containing the EGP route. You can reference this from the PROPAGATE DESTINATION filter you create.
- Create a PROPAGATE DESTINATION filter:

```
NCL> CREATE ROUTING CONTROL PROTOCOL rip-1 -
_NCL> PROPAGATE DESTINATION prop-egp-rip1 FILTER ACTION PASS

NCL> SET ROUTING CONTROL PROTOCOL rip-1 PROPAGATE DESTINATION -
_NCL> prop-egp-rip1 DOMAIN {{ROUTING NETWORK PROTOCOL IP DOMAIN -
_NCL> EGP_PASS}}

NCL> SET ROUTING CONTROL PROTOCOL rip-1 PROPAGATE DESTINATION -
_NCL> PROP-EGP-RIP1 ADDRESS MATCH TYPE EXACT

NCL> SET ROUTING CONTROL PROTOCOL rip-1 PROPAGATE DESTINATION -
_NCL> PROP-EGP-RIP1 SOURCES {ROUTING CONTROL PROTOCOL EGP-1}

NCL> ENABLE ROUTING CONTROL PROTOCOL rip-1 PROPAGATE DESTINATION -
_NCL> PROP-EGP-RIP1
```

Example 4

The two Marketing departments are associated with the DECNIS with different instances of RIP, which means that there is usually no traffic between the two departments. If you wanted to create reachability between the two Marketing departments, you could take one of the following actions:

- Associate Marketing (Leeds office) with the RIP-2 instance. This would also add reachability to the Sales office. For details about creating RIP instances, see Section 10.10.
- Set up two propagate filters on DECNIS A. One filter could be set up to propagate RIP-2 routes into RIP-A. The other filter could be set up to propagate RIP-A routes into RIP-2. This would also add reachability to the Sales office. To set up these filters, enter the following commands:

```
NCL> CREATE ROUTING CONTROL PROTOCOL RIP-A -
_NCL> PROPAGATE DESTINATION PROP-RIP2-RIPA FILTER ACTION PASS

NCL> SET ROUTING CONTROL PROTOCOL RIP-A -
_NCL> PROPAGATE DESTINATION PROP-RIP2-RIPA ADDRESS MATCH TYPE PREFIX

NCL> SET ROUTING CONTROL PROTOCOL RIP-A -
_NCL> PROPAGATE DESTINATION PROP-RIP2-RIPA SOURCES {{ROUTING CONTROL -
_NCL> PROTOCOL RIP-2}}

NCL> ENABLE ROUTING CONTROL PROTOCOL RIP-A -
_NCL> PROPAGATE DESTINATION PROP-RIP2-RIPA

NCL> CREATE ROUTING CONTROL PROTOCOL RIP-2 -
_NCL> PROPAGATE DESTINATION PROP-RIPA-RIP2 FILTER ACTION PASS

NCL> SET ROUTING CONTROL PROTOCOL RIP-2 -
_NCL> PROPAGATE DESTINATION PROP-RIPA-RIP2 ADDRESS MATCH TYPE PREFIX

NCL> SET ROUTING CONTROL PROTOCOL RIP-2 -
_NCL> PROPAGATE DESTINATION PROP-RIPA-RIP2 SOURCES {{ROUTING CONTROL -
_NCL> PROTOCOL RIP-A}}

NCL> ENABLE ROUTING CONTROL PROTOCOL RIP-2 -
_NCL> PROPAGATE DESTINATION PROP-RIPA-RIP2
```

11.10 Examples of Setting Up Route Propagation Using RIP and OSPF

11.10.1 An Example of Propagating RIP Routes into OSPF

The following steps show how to set up a PROPAGATE DESTINATION filter for OSPF:

1. Create the PROPAGATE DESTINATION filter for OSPF and name the entity:

```
NCL> CREATE ROUTING CONTROL PROTOCOL OSPF-General -
_NCL> PROPAGATE DESTINATION RIP1-pass -
_NCL> FILTER ACTION PASS
```

2. Set the PROPAGATE DESTINATION attributes (SOURCE, ANNOUNCE ROUTE ATTRIBUTES, and ADDRESS MATCH TYPE):

   ```
   NCL> SET ROUTING CONTROL PROTOCOL OSPF-General -
   _NCL>  PROPAGATE DESTINATION RIP1-pass -
   _NCL> SOURCE { ROUTING CONTROL PROTOCOL RIP1 }, -
   _NCL> ANNOUNCE ROUTE ATTRIBUTES { OSPF EXTERNAL TYPE 1 }, -
   _NCL> ADDRESS MATCH TYPE PREFIX
   ```

   Since no IP domain has been specified, all RIP1 routes in the Forwarding Table will be propagated into OSPF.

3. Enable the PROPAGATE DESTINATION filter:

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL OSPF-General -
   _NCL> PROPAGATE DESTINATION RIP1-pass
   ```

11.10.2 An Example of Propagating OSPF Routes into RIP

To propagate all OSPF routes into another protocol instance, you must set up a PROPAGATE DESTINATION filter for each OSPF route type (there are four in total).

1. Create the PROPAGATE DESTINATION entities for rip-1 and name the entities:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL rip-1 -
   _NCL> PROPAGATE DESTINATION OSPF-Intra-pass -
   _NCL> FILTER ACTION PASS, SOURCE ROUTE ATTRIBUTE { OSPF Intra Area }

   NCL> CREATE ROUTING CONTROL PROTOCOL rip-1 -
   _NCL> PROPAGATE DESTINATION OSPF-Inter-pass -
   _NCL> FILTER ACTION PASS, SOURCE ROUTE ATTRIBUTE { OSPF Inter Area }

   NCL> CREATE ROUTING CONTROL PROTOCOL rip-1 -
   _NCL> PROPAGATE DESTINATION OSPF-Extyp1-pass -
   _NCL> FILTER ACTION PASS, SOURCE ROUTE ATTRIBUTE { OSPF External Type 1 }

   NCL> CREATE ROUTING CONTROL PROTOCOL rip-1 -
   _NCL> PROPAGATE DESTINATION OSPF-Extyp2-pass -
   _NCL> FILTER ACTION PASS, SOURCE ROUTE ATTRIBUTE { OSPF External Type 2 }
   ```

2. Set the PROPAGATE DESTINATION attributes (SOURCE and ADDRESS MATCH TYPE):

```
NCL> SET ROUTING CONTROL PROTOCOL rip-1 -
_NCL> PROPAGATE DESTINATION OSPF-Intra-pass -
_NCL> SOURCE { ROUTING CONTROL PROTOCOL OSPF-Area-0 }, -
_NCL> ADDRESS MATCH TYPE PREFIX

NCL> SET ROUTING CONTROL PROTOCOL rip-1 -
_NCL> PROPAGATE DESTINATION OSPF-Inter-pass -
_NCL> SOURCE { ROUTING CONTROL PROTOCOL OSPF-Area-0 }, -
_NCL> ADDRESS MATCH TYPE PREFIX

NCL> SET ROUTING CONTROL PROTOCOL rip-1 -
_NCL> PROPAGATE DESTINATION OSPF-Extyp1-pass -
_NCL> SOURCE { ROUTING CONTROL PROTOCOL OSPF-General }, -
_NCL> ADDRESS MATCH TYPE PREFIX

NCL> SET ROUTING CONTROL PROTOCOL rip-1 -
_NCL> PROPAGATE DESTINATION OSPF-Extyp2-pass -
_NCL> SOURCE { ROUTING CONTROL PROTOCOL OSPF-General }, -
_NCL> ADDRESS MATCH TYPE PREFIX
```

Since no IP domains have been specified in any of the PROPAGATE DESTINATION filters, all OSPF routes in the Forwarding Table will be propagated into RIP.

3. Enable the PROPAGATE DESTINATION filter:

```
NCL> ENABLE ROUTING CONTROL PROTOCOL rip-1 -
_NCL> PROPAGATE DESTINATION OSPF-Intra-pass

NCL> ENABLE ROUTING CONTROL PROTOCOL rip-1 -
_NCL> PROPAGATE DESTINATION OSPF-Inter-pass

NCL> ENABLE ROUTING CONTROL PROTOCOL rip-1 -
_NCL> PROPAGATE DESTINATION OSPF-Extyp1-pass

NCL> ENABLE ROUTING CONTROL PROTOCOL rip-1 -
_NCL> PROPAGATE DESTINATION OSPF-Extyp2-pass
```

11.11 IP Domains

This section describes how to set up an IP domain.

11.11.1 Introduction

An IP domain is a set of IP address/mask pairs, specified in a DOMAIN subentity of the ROUTING NETWORK PROTOCOL IP entity. A single DOMAIN subentity can be reused by any number of RECEIVE DESTINATION, PROPAGATE DESTINATION and/or AGGREGATION entities.

Note

If you want to filter all routes or the default route, do not set up an IP domain.

11.11.2 Setting Up IP Domains

To set up an IP domain, follow these steps:

1. Create the DOMAIN entity by entering the following command:

   ```
   NCL> CREATE ROUTING NETWORK PROTOCOL IP DOMAIN domain-name
   ```

   where *domain-name* is a name you give to the domain you want to create.

2. Put the addresses and masks of the routes you want to filter in the domain by entering the following command:

   ```
   NCL> SET ROUTING NETWORK PROTOCOL IP DOMAIN domain-name -
   _NCL> IP ADDRESS {{ADDRESS=a.b.c.d, MASK=e.f.g.h} ,...}
   ```

   where *a.b.c.d* is the address of the route and *e.f.g.h* is the mask of the route.

   For example:

   ```
   NCL> SET ROUTING NETWORK PROTOCOL IP DOMAIN DOMAIN_1 -
   _NCL> IP ADDRESS {{ADDRESS=16.12.4.1, MASK=255.255.255.0}, -
   _NCL> {ADDRESS=16.13.4.1, MASK=255.255.255.0}}
   ```

11.11.3 Adding Addresses to an Existing Domain

To add addresses to an existing domain, enter the following command:

```
NCL> ADD ROUTING NETWORK PROTOCOL IP DOMAIN domain-name -
_NCL> IP ADDRESS {{ADDRESS=a.b.c.d, MASK=e.f.g.h} ,...}
```

11.11.4 IP Domains and the ADDRESS MATCH TYPE

This section describes how the setting of the ADDRESS MATCH TYPE affects which routes are filtered. You set the ADDRESS MATCH TYPE in the RECEIVE DESTINATION or PROPAGATE DESTINATION entities.

When you create an IP domain, you set it up with a list of addresses and masks of the routes you want to filter.

By default, the ADDRESS MATCH TYPE characteristic is set to PREFIX. This means that only the prefix of the address and mask of the route need to match. The prefix is the part of the address with a nonzero mask.

If the ADDRESS MATCH TYPE characteristic is set to EXACT, both the address and the mask of the route being matched must be identical to one of the address/mask pairs listed in the IP domain.

Table 11–9 shows examples of IP Domain addresses and associated route address matches where the ADDRESS MATCH TYPE characteristic is set to PREFIX.

Table 11–9 Address Matching

ADDRESS MATCH TYPE = PREFIX

IP Domain A Contains:		**Route Address Matches:**	
Address	**Mask**	**Address**	**Mask**[1]
16.36[2]	255.255	16.36.36.1	255.255.0.0
		16.36.19.3	255.255.255.0
		16.36.51.1	255.255.0.0
16.37.18	255.255.255	16.37.18.1	255.255.255.0
		16.37.18.2	255.255.255.0
		16.37.18.3	255.255.255.255
17.38.29.1	255.255.255.255	17.38.29.1	255.255.255.255

[1] Note that the mask of the route being matched must always be as long as or longer than the mask in the IP Domain.

[2] Only the prefix of the address and mask of the route need to be listed in the IP domain.

11.11.5 Overlapping IP Domains

RECEIVE DESTINATION Filters and Overlapping IP Domains

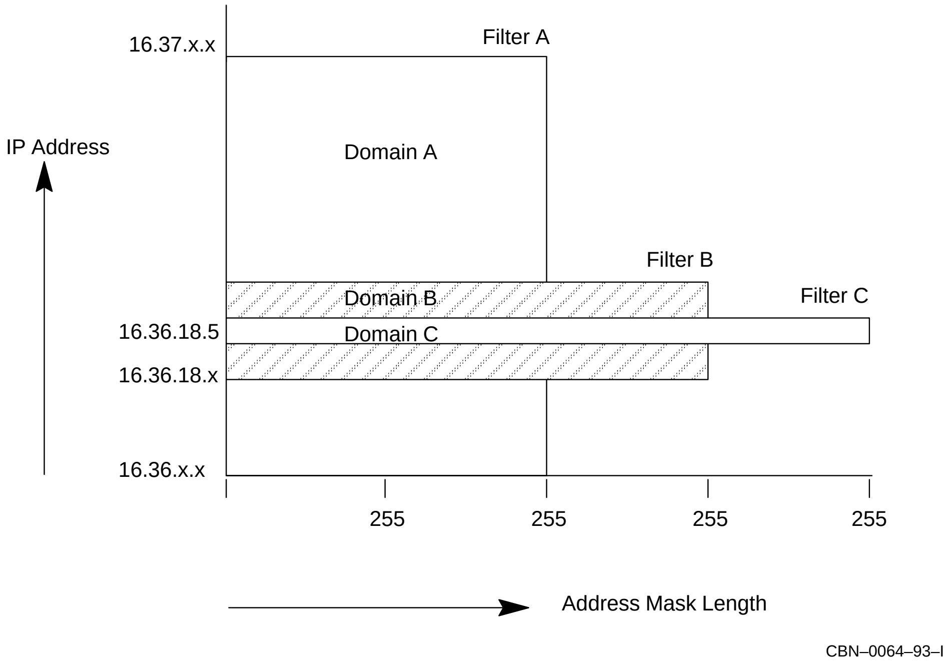
In Figure 11–10, Domain A has been set up for use with Filter A. Filter A has a filter action of BLOCK to ensure that routes 16.36.x.x are not accepted into the Routing Table. If you later decide that routes 16.36.18.x should pass, you could set up a domain (Domain B) for use with a filter (Filter B) with the filter action PASS. You could then set up another domain, Domain C, to use with Filter C and filter action BLOCK to ensure that address 16.36.18.5 is not accepted into the Routing Table.

Filters with IP domains containing the longest address masks are always checked first. As soon as an address match is found, the filter action is executed on the advertisement received.

PROPAGATE DESTINATION Filter and Overlapping IP Domains

The same domains can apply to PROPAGATE DESTINATION entities, although the filter actions would be exactly the opposite from the ones for RECEIVE DESTINATION filters. That is, Filter A's filter action would be PASS to propagate a route into another protocol instance, filter B would override that with a filter action of BLOCK, and Filter C would have another filter action of PASS.

Figure 11–10 IP Domains and Filters



12
BGP Route Propagation and Filtering

12.1 Introduction

This chapter describes how the DECNIS receives and propagates Border Gateway Protocol (BGP) routes and how you can filter these routes.

Refer to Section 10.12 for instructions on setting up a basic BGP configuration.

This chapter assumes that you have read Chapter 11, which describes received and propagated route filtering for other IP routing protocols.

12.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

12.1.2 Main Differences Between BGP and Other IP Protocols

There are a number of differences between the way BGP receives and propagates routes, and the way this is done for other protocols, as described in Chapter 11. These differences are as follows:

- By default, BGP will not do either of the following:
  - Receive routing updates, including BGP updates.
  - Propagate any routing information, including BGP routing information, to external BGP neighbors or to other protocols.

  For this reason, you must explicitly set up BGP to receive and propagate routes.
- Within an autonomous system (AS), the BGP routers must have identical policies for receiving routes. Refer to Section 12.1.3 for an explanation of policies.

- The order in which BGP filters are created is significant because it can affect the outcome of the filtering process. Each filter is by default examined in the order that it was created and the first filter in the sequence which matches is used. Refer to Section 12.6 for details.
- Unlike other routing protocols, BGP does not periodically re-advertise routes. It re-advertises routes only when there are changes to routing information.
- BGP contains additional configurable characteristics that can be advertised with routes, and used to filter received routes. This allows greater control over the way BGP routes are managed.

12.1.3 BGP Policies

The BGP protocol enforces routing **policies**. Policies are sets of criteria that determine which routes are received and propagated, and the preference assigned to them.

There are two types of BGP policy:

- **Receive policies** control which routes are accepted into the BGP Routing Table. You implement receive policies by setting up RECEIVE DESTINATION entities for BGP.
- **Send policies** control which BGP routes are advertised, and the attributes that will be applied to them when they are advertised. You implement send policies by setting up PROPAGATE DESTINATION entities for BGP.

12.2 BGP Received Route Filtering

12.2.1 Introduction

In order for the DECNIS to process the BGP routing updates it receives, you must set up route filtering for received BGP routes.

12.2.1.1 What Happens Without BGP Route Filtering?

If you do not set up BGP route filtering, then the DECNIS will not process routes advertised by neighboring BGP routers, or place them in the BGP Routing Table. This means that BGP routes will not be used for IP packet forwarding, and they will not be available to advertise.

12.2.1.2 What BGP Route Filtering Does

As well as allowing the DECNIS to receive routes, BGP route filtering determines:

- Which BGP routes are placed in the BGP Routing Table. Each routing protocol has its own Routing Table holding the routes it has accepted.
- The preference given to each route. Refer to Table 12–2 for more information.

12.2.2 How BGP Receives Routes

This section describes how BGP receives advertised routes.

12.2.2.1 BGP Route Information Advertised and Received

Figure 12–1 shows an example of a route advertisement received from a neighboring BGP router, and the information placed in the BGP Routing Table. Note that in this diagram, it is assumed that a BGP RECEIVE DESTINATION filter has been set up, and that it has passed the route to the DECNIS.

Table 12–1 explains the route advertisement information and Table 12–2 explains the Routing Table information.

Table 12–1 Advertisement Information

Information	Explanation
Address	The IP address.
Mask	The IP address mask.
Route Type	Protocol from which this route was learnt - in this case, BGP
AS path	List of autonomous systems that this information has traversed so far

Figure 12–1 Advertising and Receiving BGP Routes

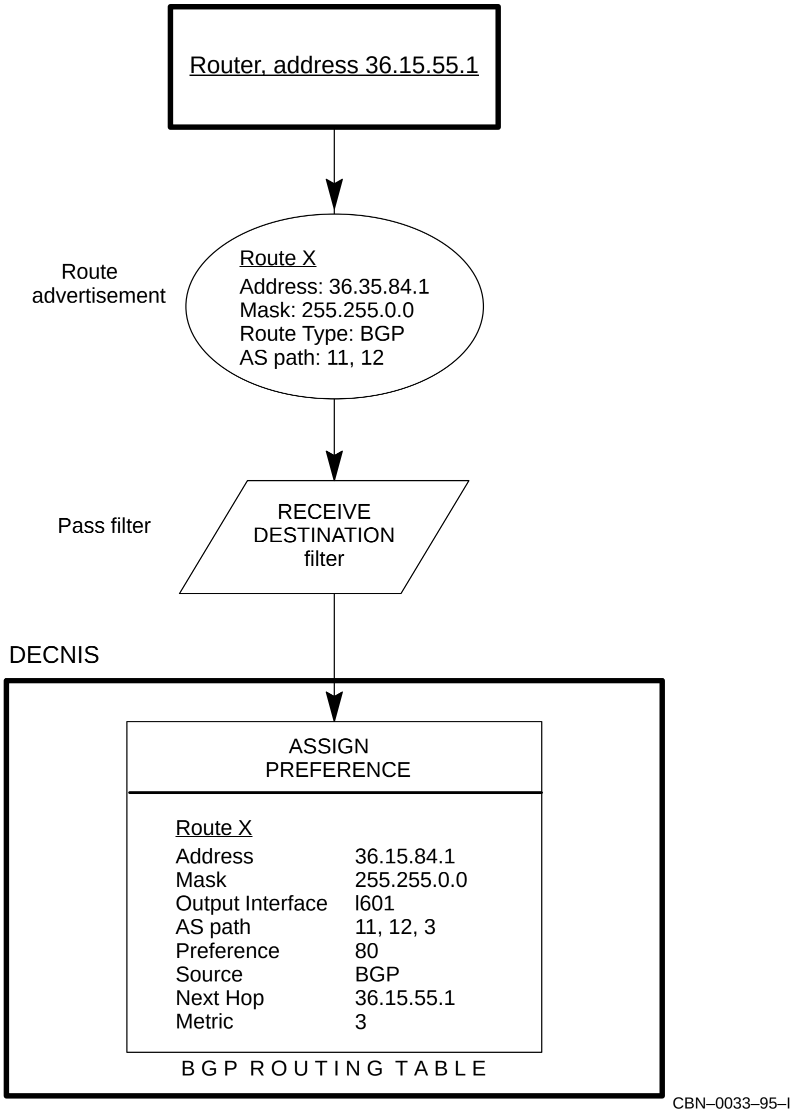


Table 12–2 Routing Table Data

Information	Explanation
Address	IP address as received from the route advertisement.
Mask	IP address mask as received from the route advertisement.
Output Interface	The DECNIS interface through which data should be sent out to this address. By definition this is the interface through which the route advertisement was received.
AS path	List of autonomous systems that the routing information has already traversed.
Preference	A number indicating the preference to be assigned to this route. The lower the number, the more preferred the route. Default preferences are assigned to each protocol. The default route preference for BGP is 80. You can change the default preference of BGP or any other routing protocol. It is recommended that you do not give any two protocols the same preference value.
Metric	BGP does not advertise a metric cost. The DECNIS assigns a metric cost to BGP routes, using the AS path length. The formula is: *Number of autonomous systems* + 1
Source	The routing protocol from which the advertisement was received.
Next Hop	The next address to which data should be forwarded after this address. This is usually the address of the router that sends the advertisement; however, BGP allows you to specify a different, forwarding address.

12.2.2.2 How the DECNIS Places Routes in the BGP Routing Table

When a BGP route advertisement arrives at the DECNIS, the DECNIS does the following:

1. Examines the route to see if it matches any of the RECEIVE DESTINATION filters for BGP.
2. If a route matches, and the filter passes it, assigns a preference and metric to the route. See Table 12–2 for details.
3. If it is the only route to a particular destination, places it in the BGP Routing Table.

   If there is more than one route to a destination, it follows the procedure in Section 12.2.2.3.

12.2.2.3 Deciding Between Multiple Routes to a Destination

BGP allows only one route to a destination to be stored in its Routing Table. If more than one route to a destination passes the RECEIVE DESTINATION filters for BGP, the DECNIS decides among them on the basis of the following criteria, in the order listed:

1. Preference. The route with the best preference is selected.
2. Metric cost. If the preferences are the same, the route with the lowest metric cost is selected.
3. External or internal neighbor. If the metric costs are the same, a route received from an external BGP neighbor is preferred over one from an internal BGP neighbor.
4. Multi exit discriminator (MED). If all routes are from External BGP neighbors, the MED is used. The MED is a number whose purpose is to act as a tie-breaker for route selection. See Section 12.5.2.2 for details.
5. IP address. If no routes have a MED, or all MEDs are the same, BGP selects the route with the lowest IP address for the neighbor it was learnt from.

12.2.2.4 Routes Passed from the Routing Tables to the Forwarding Table

Once the BGP Routing Table has accepted a route, it passes it on to the Forwarding Table. The Forwarding Table is used for IP packet forwarding and route advertising.

If the Forwarding Table receives the same route from more than one Routing Table, then it selects the route with the best preference. If they have equal preference, then it checks metric cost, and the route with the lowest metric cost is selected. For each route, the Forwarding Table can contain up to four ‘equal cost routes’, that is, routes of the same metric cost and preference. Any other versions of the route are discarded.

12.3 Setting Up Filtering of Received BGP Routes

This section describes how to configure the DECNIS to filter BGP received routes. Refer to Chapter 11 to find out how to filter received routes for all other Control Protocols.

12.3.1 Requirement for Receive Filters Within an Autonomous System

Within an autonomous system (AS), all BGP routers must have identical receive policies. This means that all the DECNIS BGP routers within an AS must have identical RECEIVE DESTINATION filters for BGP.

12.3.2 The RECEIVE DESTINATION Filter

Each RECEIVE DESTINATION for BGP has the characteristics shown in Table 12–3.

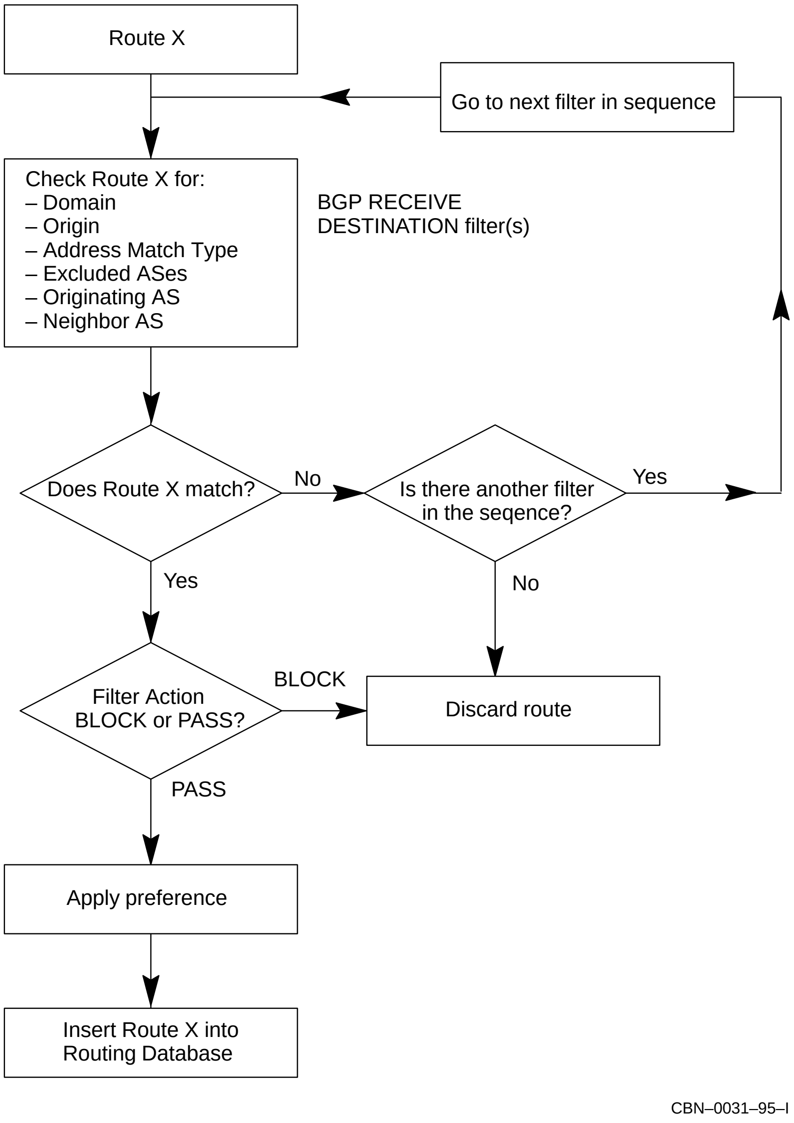
Table 12–3 RECEIVE DESTINATION Characteristics

For this characteristic...	Specify...
DOMAINS	A list of IP domains containing the addresses and subnet masks of the routes you want to filter.
ADDRESS MATCH TYPE	EXACT or PREFIX. The default is PREFIX.
EXCLUDED AUTONOMOUS SYSTEMS	A set of autonomous systems. If the path of a route includes any of the specified autonomous systems, the route will not match this filter.
ORIGINATING AUTONOMOUS SYSTEM	The number of the autonomous system (AS) where the route originated. The default is zero (the originating AS is ignored).
NEIGHBOR AUTONOMOUS SYSTEM	The number of the AS from which the route was learnt. The default is zero (the neighbor AS is ignored).
ORIGIN	Indicates the origin of the route. One of: IGP (route is interior to the AS that originated it. This means that path information is complete); EGP (route learnt from EGP); INCOMPLETE (path is not complete); ANY (any origin. This is the default).
FILTER ACTION	BLOCK or PASS
PREFERENCE	A number indicating the preference to be assigned to this route, in the range 1–255. The default for BGP is 80. See Table 12–2 for more information.

All characteristics except FILTER ACTION and PREFERENCE are used to check whether the incoming route advertisement matches the RECEIVE DESTINATION filter. If it does, then FILTER ACTION and/or any PREFERENCE are applied to the route.

Figure 12–2 shows how BGP filters received routes.

Figure 12–2 BGP Received Route Filtering



12.3.3 Setting Up a BGP RECEIVE DESTINATION Filter

This section describes how to set up a BGP RECEIVE DESTINATION filter. Refer to Section 10.12.5 to find out how to set up a simple BGP RECEIVE DESTINATION to pass all received BGP routes.

12.3.3.1 Before You Begin

Before you set up the RECEIVE DESTINATION filter, be sure that you do the following:

- Create a NETWORK PROTOCOL entity of IP, if this has not been done already. Enter the following command:

  ```
  NCL> CREATE ROUTING NETWORK PROTOCOL IP
  ```

- Set up the list of addresses you want to block from or insert into the BGP Routing Table. To do this, you set up one or more IP domains.

  An IP domain is a set of IP address/mask pairs. You can create any number of domains. Refer to Section 11.11 to find out how to set up an IP domain.

- Create a ROUTING CONTROL PROTOCOL of type BGP, as described in Section 10.12. However, do not enable the CONTROL PROTOCOL until after you have enabled the RECEIVE DESTINATION filter.

12.3.3.2 Procedure

This section assumes that you have created a ROUTING CONTROL PROTOCOL with the name bgp. Follow these steps:

1. Create the RECEIVE DESTINATION filter and specify the filter action:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL bgp RECEIVE -
   _NCL> DESTINATION filter_name FILTER ACTION filter_action
   ```

   where *filter_action* is either PASS or BLOCK.

   Note that RECEIVE DESTINATION filters for BGP are applied in sequential order. By default, this is the order in which they are created, although you can change the sequence of the filters if you wish. Refer to Section 12.6 for details.

2. Set the address match type:

   ```
   NCL> SET  ROUTING CONTROL PROTOCOL bgp RECEIVE DESTINATION -
   _NCL> filter_name ADDRESS MATCH TYPE match_type
   ```

   where *match_type* is either EXACT or PREFIX.

   For an explanation of this attribute, see Section 11.11.4.

3. Specify the IP domain(s) with the routes that you want to filter. You can specify one IP domain or a set of IP domains.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp_1 RECEIVE DESTINATION -
   _NCL> filter-name DOMAINS {{ROUTING NETWORK -
   _NCL> PROTOCOL IP DOMAIN domain-name} ,...}
   ```

   For example:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp RECEIVE DESTINATION -
   _NCL> filter_1 DOMAINS {{ROUTING NETWORK PROTOCOL IP -
   _NCL> DOMAIN domain_1} , {ROUTING NETWORK PROTOCOL IP -
   _NCL> DOMAIN domain_2}}
   ```

   If you want this filter to apply all routes or the default route, do not specify any IP domains; see Section 11.5.7 and Section 11.5.8, respectively.

4. Specify the origin of the route. This lets you block or pass routes depending on the origin and completeness of the path information. Refer to Table 12–3 for the meaning of ORIGIN.

   ```
   NCL> SET  ROUTING CONTROL PROTOCOL bgp RECEIVE -
   _NCL> DESTINATION filter_name ORIGIN origin
   ```

   where *origin* is one of: IGP, EGP, INCOMPLETE, ANY.

5. Specify the preference of the route by entering the following command:

   ```
   NCL> SET  ROUTING CONTROL PROTOCOL bgp RECEIVE -
   _NCL> DESTINATION filter_name PREFERENCE preference
   ```

   where *preference* is the preference value. The default value of all BGP routes is 80.

6. Specify any excluded autonomous systems. If the DECNIS receives a route that has traversed one of the specified autonomous systems (that is, the autonomous system is in its AS path), the route will not match this filter.

   ```
   NCL> SET  ROUTING CONTROL PROTOCOL bgp RECEIVE -
   _NCL> DESTINATION EXCLUDED AUTONOMOUS SYSTEMS {as-number_1,...,as_number_n}
   ```

   where *as-number_1* to *as-number_n* are the numbers of the autonomous systems to be excluded.

7. Specify the number of the autonomous system (AS) that originated the route. This lets you pass or block routes from that AS.

   ```
   NCL> SET  ROUTING CONTROL PROTOCOL bgp RECEIVE -
   _NCL> DESTINATION ORIGINATING AUTONOMOUS SYSTEM as_number
   ```

   where *as_number* is the number of the AS that originated the route.

8. Specify the number of the AS from which the route was learnt. This lets you pass or block routes received from that AS.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp RECEIVE DESTINATION -
   _NCL> NEIGHBOR AUTONOMOUS SYSTEM as_number
   ```

   where *as_number* is the AS number.

9. Enable the RECEIVE DESTINATION filter:

   ```
   NCL> ENABLE  ROUTING CONTROL PROTOCOL bgp -
   _NCL> RECEIVE DESTINATION filter_name
   ```

12.4 BGP Route Propagation

You must explicitly set up BGP route propagation in order for BGP to propagate routes to external BGP neighbors.

The DECNIS by default propagates all BGP routes to internal BGP neighbors.

12.4.1 What Happens Without BGP Route Propagation?

If you do not set up BGP route propagation, then the DECNIS will not propagate:

- Any routes learnt via BGP to external BGP neighbors.
- Any routes from Interior Gateway Protocols (IGP) (such as RIP) to BGP.

12.4.2 What BGP Route Propagation Does

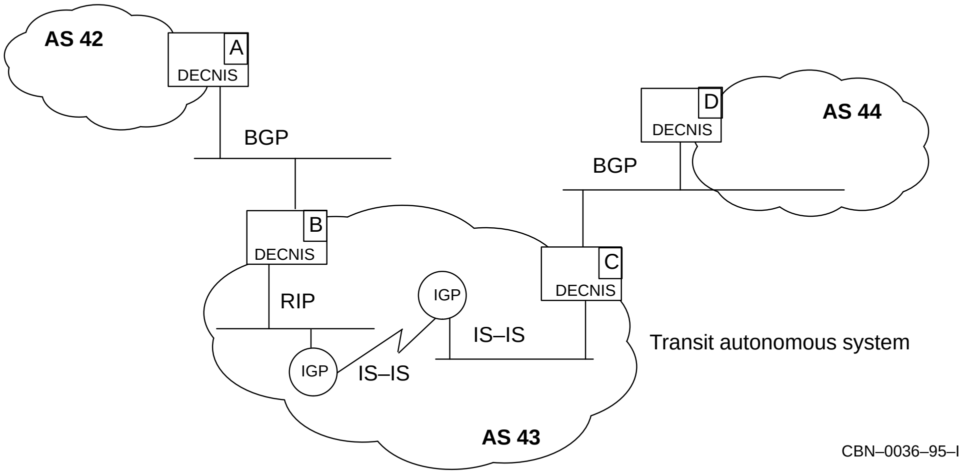
BGP route propagation determines:

- Which BGP routes are propagated to external BGP neighbors.
- Which routes learnt from other protocols are propagated into BGP.
- The autonomous systems to which routes should be sent.
- Information that should be sent with the route - for example, you can specify the next hop.

12.4.3 Transit Autonomous Systems

Where there are more than two autonomous systems connected together, routes can be passed through one or more of them before they reach their destination. The autonomous systems which pass on routes are known as **transit** autonomous systems. Within a transit AS, routes can be received by one BGP router and passed to another, either internal or external.

Figure 12–3 Route Propagation



For example, in Figure 12–3, AS 43 is a transit AS.

12.4.4 Advertising Routes

By default, all Control Protocols match BGP Propagate filters. This is because the default for the SOURCES attribute (which specifies Control Protocols to be propagated or blocked) is ALL Control Protocols.

If you only want to propagate a limited set of protocols into BGP, you must specify the Control Protocols you want in the SOURCES attribute.

Note that this applies even if you want to propagate BGP routes back into BGP. For example, in Figure 12–3, if you want all the DECNIS routers to propagate BGP routes, but not other routes, then the Propagate filters on each router must specify the name of the BGP Control Protocol entity in the SOURCES attribute.

12.4.5 Advertising Routes to Other Routing Protocols

In a transit AS, BGP needs to propagate BGP routes into an IGP to ensure that a path for the routes exists inside the AS.

BGP automatically propagates routes to its internal BGP neighbors. However, when the internal neighbor receives the route, it does not install it in its Forwarding Table. Instead, it waits for the route to be received by an IGP. Thus, a DECNIS BGP router will not propagate a BGP route received from an internal BGP neighbor until it sees that it has also received it from an IGP.

This ensures that a BGP route is only advertised to an external BGP neighbor when all routers in the AS have the route to the external BGP neighbor.

In Figure 12–3, DECNIS B and DECNIS C are internal BGP neighbors in AS 43. DECNIS B has the External BGP neighbor DECNIS A; DECNIS C has the External BGP neighbor DECNIS D.

If you want to propagate routes across AS 43—for example, from DECNIS D to DECNIS A—you must set up DECNIS B and DECNIS C to propagate BGP routes into IGPs.

For example, on DECNIS B, you would create an IS–IS propagate filter to propagate routes from the BGP routing protocol into IS–IS.

12.5 Setting Up BGP Route Propagation

12.5.1 The PROPAGATE DESTINATION Filter

Each PROPAGATE DESTINATION filter has the characteristics listed in Table 12–4.

Table 12–4 PROPAGATE DESTINATION Characteristics

For this characteristic...	Specify...
DOMAINS	A list of IP domains containing the addresses and subnet masks of the routes to be propagated or blocked.
ADDRESS MATCH TYPE	EXACT or PREFIX. The default is PREFIX.
SOURCES	The names of the CONTROL PROTOCOL entities whose routes you want to propagate. In order to match, the route must come from one of the specified Control Protocols. The default is an empty set ({ }), which matches all Control Protocols.
SOURCE ROUTE ATTRIBUTES	The type of route which you want to filter. Only used if one of the Control Protocols specified in SOURCES is of the type IS–ISLevel2 or OSPF.
EXCLUDED AUTONOMOUS SYSTEMS	A set of autonomous systems. If the path of a route includes any of the specified autonomous systems, the route will not match this filter.
ORIGINATING AUTONOMOUS SYSTEM	Number of the autonomous system (AS) where the route originated. The default is zero (the originating AS is ignored).
NEIGHBOR AUTONOMOUS SYSTEM	The number of the AS from which the route was learnt. The default is zero (the neighbor AS is ignored).
ORIGIN	Indicates the origin of the route. One of: IGP (route is interior to the AS that originated it. This means that the path information is complete); EGP (route learnt from EGP), INCOMPLETE (path is not complete), ANY (any origin. This is the default).
FILTER ACTION	BLOCK or PASS
ANNOUNCE NEXT HOP	The IP address of the router you want to use for the next hop to the destination. This does not have to be the address of the router that sent the route, but it must be in the same subnet as the router that sent the route.
ANNOUNCE NEIGHBORS	The numbers of the autonomous systems to which the route matching this PROPAGATE DESTINATION is to be sent. The default is to propagate to all autonomous systems.
MULTI EXIT DISCRIMINATOR	A value used as a tie-breaker by receiving BGP routers when two or more routes exist to a particular destination.
DISCRIMINATOR TYPE	Specifies how the MULTI EXIT DISCRIMINATOR is advertised. One of the following: NONE; FIXED; PASSTHROUGH. Refer to Section 12.5.2.2, step 11.

The first eight characteristics are used to check whether the route matches the PROPAGATE DESTINATION filter. If it does, it will be treated in the way specified by the FILTER ACTION. If the FILTER ACTION is PASS, then the route will be advertised with the last four characteristics applied (if specified).

Figure 12–4 shows the route filtering process for BGP route propagation.

12.5.2 Setting Up a BGP PROPAGATE DESTINATION Filter

This section describes how to set up a BGP PROPAGATE DESTINATION filter. Refer to Section 10.12.5 to find out how to set up a simple PROPAGATE DESTINATION filter to propagate all BGP routes.

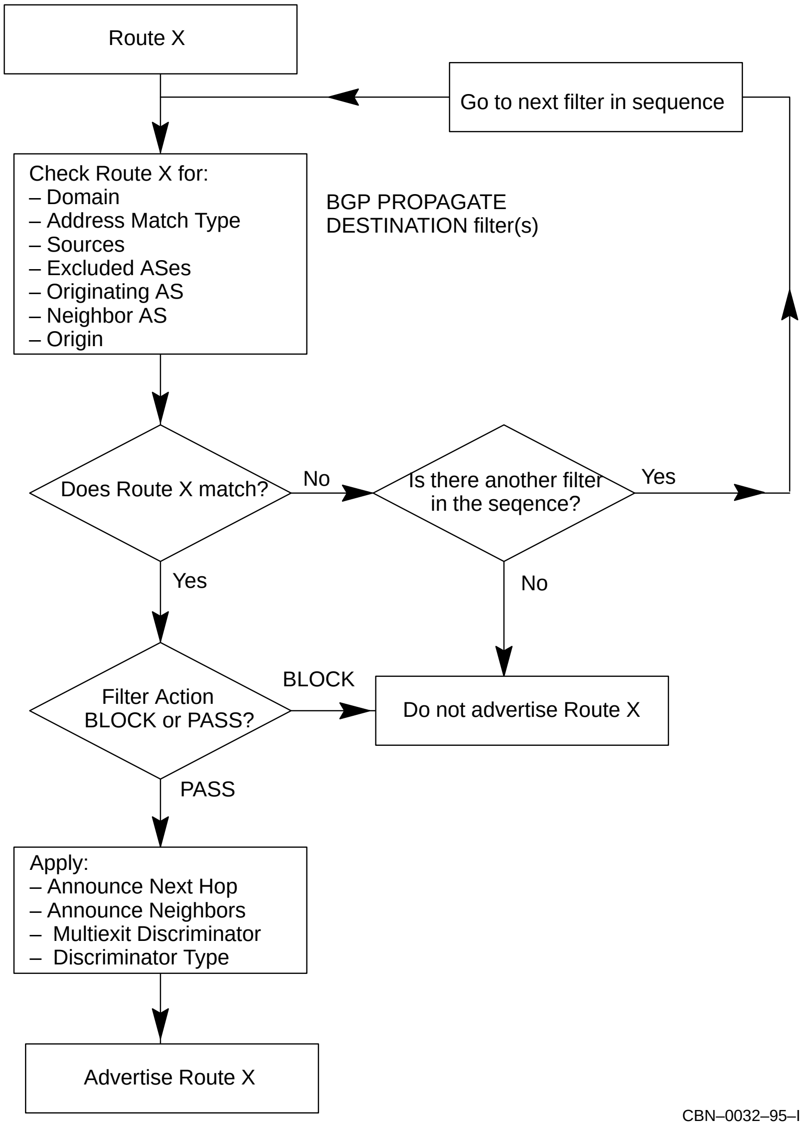
12.5.2.1 Before You Begin

Before you set up the PROPAGATE DESTINATION filter, be sure that you do the following:

- Create a NETWORK PROTOCOL entity of IP, if this has not been done already. Refer to Section 12.3.3.1 for instructions.
- Set up the list of addresses you want to propagate, or block from being propagated. To do this, you set up one or more IP domains.

  An IP domain specifies a set of IP address/mask pairs. You can create any number of domains. Refer to Section 11.11 to find out how to set up an IP domain.
- Create a ROUTING CONTROL PROTOCOL of type BGP, as described in Section 10.12. However, do not enable the CONTROL PROTOCOL until after you have enabled the PROPAGATE DESTINATION filter.
- If you want BGP to propagate to other Control Protocols, set the PREFERENCE attribute in the ROUTING CONTROL PROTOCOL for BGP to be better than the preferences set in the other ROUTING CONTROL PROTOCOLS. Refer to Section 12.8 for details.

Figure 12–4 BGP Route Advertising



12.5.2.2 Procedure

This section assumes that you have created a ROUTING CONTROL PROTOCOL with the name bgp. Follow these steps:

1. Create the PROPAGATE DESTINATION filter, and specify the filter action.

   ```
   NCL> CREATE  ROUTING CONTROL PROTOCOL bgp -
   _NCL> PROPAGATE DESTINATION filter-name FILTER ACTION filter-action
   ```

   where *filter-name* is the name of the PROPAGATE DESTINATION entity, and *filter-action* is either PASS or BLOCK.

   Note that PROPAGATE DESTINATION filters for BGP are applied in sequential order. By default, this is the order in which they are created, although you can change the sequence of the filters if you wish. Refer to Section 12.6 for details.

2. Specify the IP domains containing the routes you want to propagate by entering the following command:

   ```
   NCL> SET  ROUTING CONTROL PROTOCOL bgp PROPAGATE -
   _NCL> DESTINATION filter_name DOMAINS -
   _NCL> {{ROUTING NETWORK PROTOCOL domain-name}, .....}
   ```

   where *domain-name* is the name of the domain.

3. Set the address match type:

   ```
   NCL> SET  ROUTING CONTROL PROTOCOL bgp PROPAGATE DESTINATION -
   _NCL> prop_name ADDRESS MATCH TYPE match_type
   ```

   where *match_type* is either EXACT or PREFIX.

   For an explanation of this attribute, see Section 11.11.4.

4. Use the SOURCES attribute to specify the names of the CONTROL PROTOCOL instances that you want to propagate from—that is, the protocol instances which learnt the routes that you want BGP to advertise. If you do not specify SOURCES, or specify it with no control protocol names, then all Control Protocols will match.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-prot-name -
   _NCL> PROPAGATE DESTINATION name SOURCES -
   _NCL> {{ROUTING CONTROL PROTOCOL source-name}, ....}
   ```

   where *source-name* is the name of a CONTROL PROTOCOL entity.

For example, to propagate BGP and Local Routes to BGP, you could enter the following:

```
NCL> SET  ROUTING CONTROL PROTOCOL bgp PROPAGATE -
_NCL> DESTINATION bgp_prop SOURCES -
_NCL> {{ROUTING CONTROL PROTOCOL LOCAL}, {ROUTING CONTROL PROTOCOL bgp}}
```

Note that you need to explicitly propagate BGP routes into BGP (unless you are propagating from all Control Protocols).

5. Specify any excluded autonomous systems. Routes that have traversed one of the specified autonomous systems will not match this filter.

```
NCL> SET ROUTING CONTROL PROTOCOL bgp PROPAGATE DESTINATION -
_NCL> filter-name EXCLUDED AUTONOMOUS SYSTEMS {as_number_1,..,as_number_n}
```

   where *as_number_1*, and so on, are the list of autonomous system numbers.

6. You can the neighboring autonomous systems to which routes should be advertised. The default is for routes to be advertised to all autonomous systems. Enter the following command:

```
NCL> SET  ROUTING CONTROL PROTOCOL BGP PROPAGATE -
_NCL> DESTINATION ANNOUNCE NEIGHBORS {as_number_1,...,as_number_n}
```

   where *as_number_1* to *as_number_n* are the autonomous system numbers.

7. You can specify the number of the autonomous system (AS) that originated the route. This lets you pass or block the propagation of routes from that AS.

```
NCL> SET  ROUTING CONTROL PROTOCOL bgp PROPAGATE DESTINATION -
_NCL> prop-name ORIGINATING AUTONOMOUS SYSTEM as_number
```

   where *as_number* is the autonomous system number.

8. You can specify the number of the AS from which the route was learnt. This lets you pass or block propagation of routes received from that AS.

```
NCL> SET  ROUTING CONTROL PROTOCOL bgp PROPAGATE DESTINATION -
_NCL> prop-name NEIGHBOR AUTONOMOUS SYSTEM as_number
```

   where *as_number* is the neighboring AS number.

9. You can specify the ORIGIN of the route. This lets you block or pass routes depending on the origin and completeness of the path information. Refer to Table 12–4 for the meaning of ORIGIN.

```
NCL> SET  ROUTING CONTROL PROTOCOL bgp PROPAGATE -
_NCL> DESTINATION prop_name ORIGIN origin
```

   where *origin* is one of: IGP, EGP, INCOMPLETE, ANY.

10. You can specify the BGP router to be used as the next hop. This does not have to be the address of the router that sent the route, although it must be in the same subnet as the router that sent the route. Enter the following command:

    ```
    NCL> SET ROUTING CONTROL PROTOCOL BGP PROPAGATE -
    _NCL> DESTINATION ANNOUNCE NEXT HOP ip_address
    ```

    where *ip_address* is the IP address of the router to be used as the next hop for matching routes.

11. If you wish, specify a multi exit discriminator. This can be used as a tie-breaker by routers receiving this route (see Section 12.2.2.3 for details).

    You can do any of the following:

    - Set a value for the multi exit discriminator. Enter the following command:

      ```
      NCL> SET ROUTING CONTROL PROTOCOL bgp PROPAGATE DESTINATION -
      _NCL> prop-name DISCRIMINATOR TYPE FIXED -
      _NCL> MULTI EXIT DISCRIMINATOR discrim_value
      ```

      where *discrim_value* is a value in the range 0–65535.

    - Set the metric cost of the route as the MED. Enter the following command:

      ```
      NCL> SET ROUTING CONTROL PROTOCOL bgp PROPAGATE DESTINATION -
      _NCL> prop-name DISCRIMINATOR TYPE PASSTHROUGH
      ```

    - Not send a MED. If you do not want to send a MED, either omit the attributes MULTI EXIT DISCRIMINATOR and DISCRIMINATOR TYPE altogether, or enter the following command:

      ```
      NCL> SET ROUTING CONTROL PROTOCOL bgp PROPAGATE DESTINATION -
      _NCL> prop-name DISCRIMINATOR TYPE NONE
      ```

12. Enable the PROPAGATE DESTINATION filter:

    ```
    NCL> ENABLE  ROUTING CONTROL PROTOCOL bgp -
    _NCL> PROPAGATE DESTINATION prop_name
    ```

12.6 BGP Filter Sequences

BGP filters are examined in sequential order. The first filter in the sequence which matches is used.

By default, the sequential order is the same as the order in which the filters were created. However, you can change the order, as described in Section 12.6.3.

12.6.1 How Filter Sequence Works

When you set up a BGP RECEIVE or PROPAGATE DESTINATION filter, it is assigned a sequence number that indicates its position in the list of such filters existing for BGP. For example, if you first create a PROPAGATE DESTINATION filter called BGP1, and then create one called BGP2, BGP1 will be examined first, and BPG2 next.

The order in which filters are applied can determine which routes are accepted or propagated. This means that if you create more than one Receive or Propagate filter, you must be careful to ensure that you will get the desired result.

For example, if you want to propagate IP routes from 16.1.x.x but block other IP routes from 16.x.x.x you must first create a PROPAGATE DESTINATION filter to pass 16.1.x.x and then create one to block 16.x.x.x. If you created the filters in the opposite order, the first would block IP addresses of 16.1.x.x.

12.6.2 Displaying the Sequence

To display the existing sequence of filters, enter one of the following commands:

```
NCL> SHOW ROUTING CONTROL PROTOCOL bgp -
_NCL> RECEIVE DESTINATION * SEQUENCE
```

```
NCL> SHOW ROUTING CONTROL PROTOCOL bgp -
_NCL> PROPAGATE DESTINATION * SEQUENCE
```

The sequence number for each filter will be displayed.

12.6.3 Changing the Sequence

You can change the sequential order of filters so that they will be examined in a different order from the one in which they were created. This section describes how to do this.

12.6.3.1 Example Procedure

You have created the following:

- The ROUTING CONTROL PROTOCOL bgp
- Two RECEIVE DESTINATION filters for BGP, named bgp1 and bgp2, created in that order.

To change the ordering of bgp1 and bgp2, enter either of the following commands:

```
NCL> MOVE ROUTING CONTROL PROTOCOL bgp -
_NCL> RECEIVE DESTINATION bgp2 BEFORE bgp1
```

```
NCL> MOVE ROUTING CONTROL PROTOCOL bgp -
_NCL> RECEIVE DESTINATION bgp1 AFTER bgp2
```

If bgp1 and bgp2 were PROPAGATE DESTINATION filters, created in that order, you could enter either of the following commands:

```
NCL> MOVE ROUTING CONTROL PROTOCOL bgp PROPAGATE -
_NCL> DESTINATION bgp2 BEFORE bgp1
```

```
NCL> MOVE ROUTING CONTROL PROTOCOL bgp PROPAGATE -
_NCL> DESTINATION bgp1 AFTER bgp2
```

12.6.4 Inserting a New Filter in an Existing Sequence

To create a new filter, and insert it into an existing sequence, follow these steps:

1. Create the filter as usual. This automatically adds it to the end of the sequence.
2. Use the MOVE command described in Section 12.6.3.1 to place it before or after the required filter in the existing sequence.

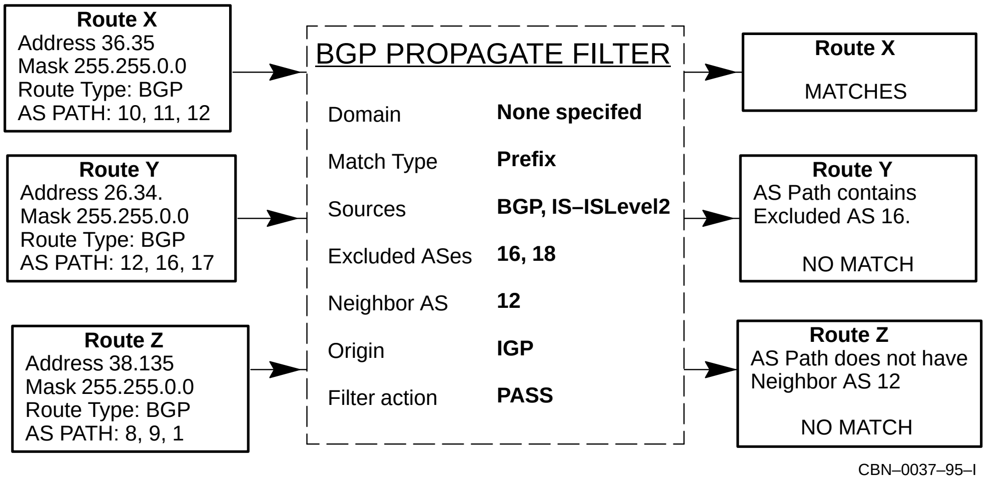
12.7 How BGP Filter Characteristics Are Examined

12.7.1 Satisfying All Characteristics in a Filter

A BGP route must satisfy all the characteristics in a given filter in order to match that filter. For example, in Figure 12–5, route Z has a route type that matches the SOURCE (BGP). However, route Z does not have the NEIGHBOR AS 12 in its AS PATH. This means that it does not match the filter.

If you want to accept routes matching either of two characteristics, you must create two different filters. For example, if you want to propagate **either** those routes that originated in AS 43 **or** those that were learnt from the neighboring AS 45, you would set up two PROPAGATE DESTINATION entities: one specifying ORIGINATING AUTONOMOUS SYSTEM 43 and the other specifying NEIGHBOR AUTONOMOUS SYSTEM 45.

Figure 12–5 Matching Filter Characteristics



12.7.2 Examining Multiple Filters

When BGP applies receive or propagate filtering, it examines each filter in turn to see if there is a match. As soon as BGP finds a match, it looks at the filter action specified, and depending on what is specified, passes or blocks the route.

If the route does not match any of the filters, it is discarded.

12.8 Changing Protocol Preferences

In general, you should always propagate from a Control Protocol with a better preference to a Control Protocol with a worse one (that is, from a Control Protocol with a lower numbered preference to a Control Protocol with a higher numbered one). For this reason, if you want to propagate BGP routes into an IGP—for example, OSPF—then BGP must have a better preference than the IGP. If the IGP has a better default preference than BGP, you will need to change either the BGP preference or the IGP preference.

For example, if you want to propagate BGP routes into OSPF External Type 2, you do not need to make any changes, as the default BGP preference of 80 is better than the OSPF External Type 2 preference of 100. However, if you want to propagate BGP routes into OSPF External Type 1, you must change one of the preferences, as the default BGP preference is worse than the OSPF External Type 1 preference of 60.

12.8.1 Default Preferences

Refer to Table 11–3 to see the default preference for all Routing Control Protocols on the DECNIS.

12.8.2 Restriction on Changing Local Preference

The preference for Local routes must always be the best. Do not change any other preference to be the same as Local.

You cannot change the preference for Local routes, because you cannot set up Receive or Propagate filters for Local routes.

12.8.3 Procedure

To change the BGP preference, follow these steps:

1. Display the preferences of the Control Protocols that you want BGP to propagate to:

   ```
   NCL> SHOW ROUTING CONTROL PROTOCOL * PREFERENCE
   ```

   This displays the preferences of all Control Protocols.

2. Change the BGP preference so that it is better than the others.

   Note

   The lower the number, the better the preference.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp-name -
   NCL> PREFERENCE preference-number
   ```

12.8.4 Example

The following Control Protocols are set up on your DECNIS. You want BGP to propagate to all of them. The BGP Control Protocol has the name bgp.

Control Protocol Type	Preference
Local	0
IS–IS Level 1	20
IP RIP	70
BGP	75
IS–IS Level 2 External	90

To give BGP the best preference, enter the following command:

```
NCL> SET ROUTING CONTROL PROTOCOL bgp PREFERENCE 1
```

Note that you must not make the BGP preference the same as Local.

12.9 Displaying the Routes in the BGP Routing Table

Once you have set up and started BGP, you can display the BGP Routing Table.

12.9.1 Procedure

To display the contents of the BGP Routing Table, enter the following command:

```
NCL> SHOW NODE decnis ROUTING CONTROL PROTOCOL bgp ROUTE * ALL
```

12.9.2 Contents of the BGP Routing Table

When you enter the command in Section 12.9.1, you will see two types of routes:

- Routes learnt from external and internal BGP neighbors. These routes will show a non-zero IP address for the `Neighbor`.
- Non-BGP routes propagated into BGP. These routes will show the IP address 0.0.0.0 for the `Neighbor`.

12.9.3 Example Display: Routes Learnt from BGP

An example of output from the command in Section 12.9.1 is shown below. It displays a route learnt from a BGP neighbor.

```
Node rtr192 Routing Control Protocol bgp Route
   [
   Address =      192.208.32.0 ,
   Mask =      255.255.255.0
   ]
AT 1995-08-17-12:52:17.372+00:00I-----

Identifiers

    Name                             =
       [
       Address =      192.208.32.0 ,
       Mask =      255.255.255.0
       ]
```

```
Status
    Paths                                =
       {
          [
          Best =          True ,
          Neighbor =          16.39.80.199 ,
          Preference =          80 ,
          Origin =          Incomplete ,
          Next Hop =          16.39.80.199 ,
          Multi Exit Discriminator =          20030 ,
          Local Preference =          0 ,
          Atomic Aggregate =          False ,
          Aggregator =          0.0.0.0 ,
          Path =
```

12.9.4 Example Display: Routes Propagated into BGP

An example of output from the command in Section 12.9.1 is shown below. It displays a non-BGP route propagated into BGP.

```
Node nis180 Routing Control Protocol bgp Route
   [
   Address =       16.36.0.0 ,
   Mask =       255.255.255.0
   ]
AT 1995-08-17-12:49:14.802+00:00I-----

Identifiers

    Name                                 =
       [
       Address =       16.36.0.0 ,
       Mask =       255.255.255.0
       ]

Status

    Paths                                =
       {
          [
          Best =          True ,
          Neighbor =          0.0.0.0 ,
          Preference =          30 ,
          Origin =          IGP ,
          Next Hop =          16.36.16.180 ,
          Multi Exit Discriminator =          0 ,
          Local Preference =          0 ,
          Atomic Aggregate =          False ,
          Aggregator =          0.0.0.0 ,
          Path =
```

12.9.5 When No BGP Routes Are Best Routes

If the display shows a number of BGP paths for a particular route, but none of the paths show `Best = True`, this indicates that there is a better route learnt by a protocol other than BGP in the Forwarding Table, and that this route is not being propagated into BGP.

To see the routes that are being propagated, that is, those in the Forwarding Table, enter the following command:

```
NCL> SHOW ROUTING IP DESTINATION ADDRESS * ALL
```

12.10 Examples of BGP Route Propagation and Filtering

This section contains examples showing how to set up BGP filters to control the way BGP receives and propagates routes.

12.10.1 Preventing an Autonomous System from Acting as a Transit AS

Figure 12–6 shows three connected autonomous systems: AS 42, AS 43 and AS 44.

- DECNIS A , B, C and D route BGP.
- DECNIS B also routes RIP.
- DECNIS C also routes Integrated IS-IS.

What You Want to Achieve

You want to prevent AS 43 from acting as a transit AS. In other words, you do not want AS 43 to pass data or routes between AS 42 and AS 44. However, you do want AS 43 to be able to receive routes from and propagate routes to AS 42 and AS 44.

General Method

On DECNIS B and DECNIS C, set up two PROPAGATE DESTINATION entities:

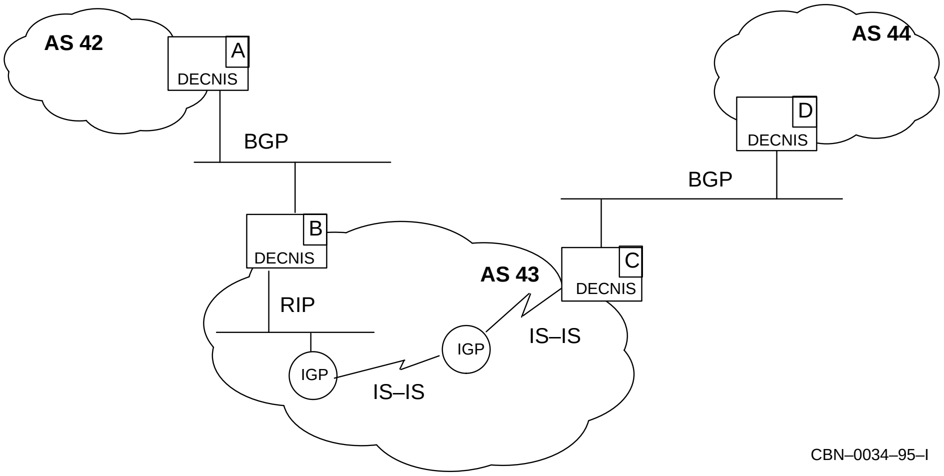
- One to prevent propagation of BGP routes to BGP external neighbors.
- One to allow propagation of IGP routes, such as RIP and IS–IS.

The Configuration So Far

The following entities have been set up on DECNIS B and DECNIS C.

DECNIS B	DECNIS C
ROUTING NETWORK PROTOCOL IP	ROUTING NETWORK PROTOCOL IP
ROUTING CONTROL PROTOCOL bgp	ROUTING CONTROL PROTOCOL bgp
ROUTING CONTROL PROTOCOL bgp NEIGHBOR as42	ROUTING CONTROL PROTOCOL bgp NEIGHBOR as44
ROUTING CONTROL PROTOCOL is-islevel2	ROUTING CONTROL PROTOCOL is-islevel2
ROUTING CONTROL PROTOCOL rip_a	

Figure 12–6 Example: Blocking Transit Traffic



12.10.1.1 Procedure

Follow these steps on both DECNIS B and DECNIS C:

1. Disable the ROUTING CONTROL PROTOCOL bgp, as follows:

   ```
   NCL> DISABLE ROUTING CONTROL PROTOCOL bgp
   ```

2. Create a PROPAGATE DESTINATION entity for BGP, and specify the filter action as BLOCK.

   ```
   NCL> CREATE  ROUTING CONTROL PROTOCOL bgp -
   _NCL> PROPAGATE DESTINATION block_bgp FILTER ACTION BLOCK
   ```

3. For the SOURCES attribute, specify the CONTROL PROTOCOL bgp. This will prevent DECNIS B from propagating BGP routes from DECNIS A, and DECNIS C from propagating BGP routes from DECNIS D.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp PROPAGATE DESTINATION -
   _NCL>bgp_block SOURCES {{ROUTING CONTROL PROTOCOL bgp}}
   ```

4. Now, create another PROPAGATE DESTINATION entity for BGP and specify the filter action as PASS:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL bgp -
   _NCL> PROPAGATE DESTINATION pass_igp FILTER ACTION PASS
   ```

5. Specify SOURCES, but do not name any Control Protocols. This will enable DECNIS B and C to propagate all non-BGP routes—for example, local, RIP, IS–IS.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp -
   _NCL> PROPAGATE DESTINATION pass_igp SOURCES {{ }}
   ```

6. Enable the PROPAGATE DESTINATION entities:

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL bgp PROPAGATE DESTINATION bgp_block
   NCL> ENABLE ROUTING CONTROL PROTOCOL bgp PROPAGATE DESTINATION igp_pass
   ```

7. Enable the ROUTING CONTROL PROTOCOL bgp:

   ```
   NCL> ENABLE ROUTING CONTROL PROTOCOL bgp
   ```

Note that in order to bring about the desired effect, the PROPAGATE DESTINATION bgp_block must be first in the sequence of PROPAGATE DESTINATION filters. Refer to Section 12.6 for more information.

12.10.2 Filtering Routes Received from Particular Autonomous Systems

Figure 12–7 shows interconnected autonomous systems. All the DECNIS routers are running BGP

You are configuring DECNIS B and C in AS 43.

What You Want To Achieve

In AS 43, you want to configure DECNIS B and C so that:

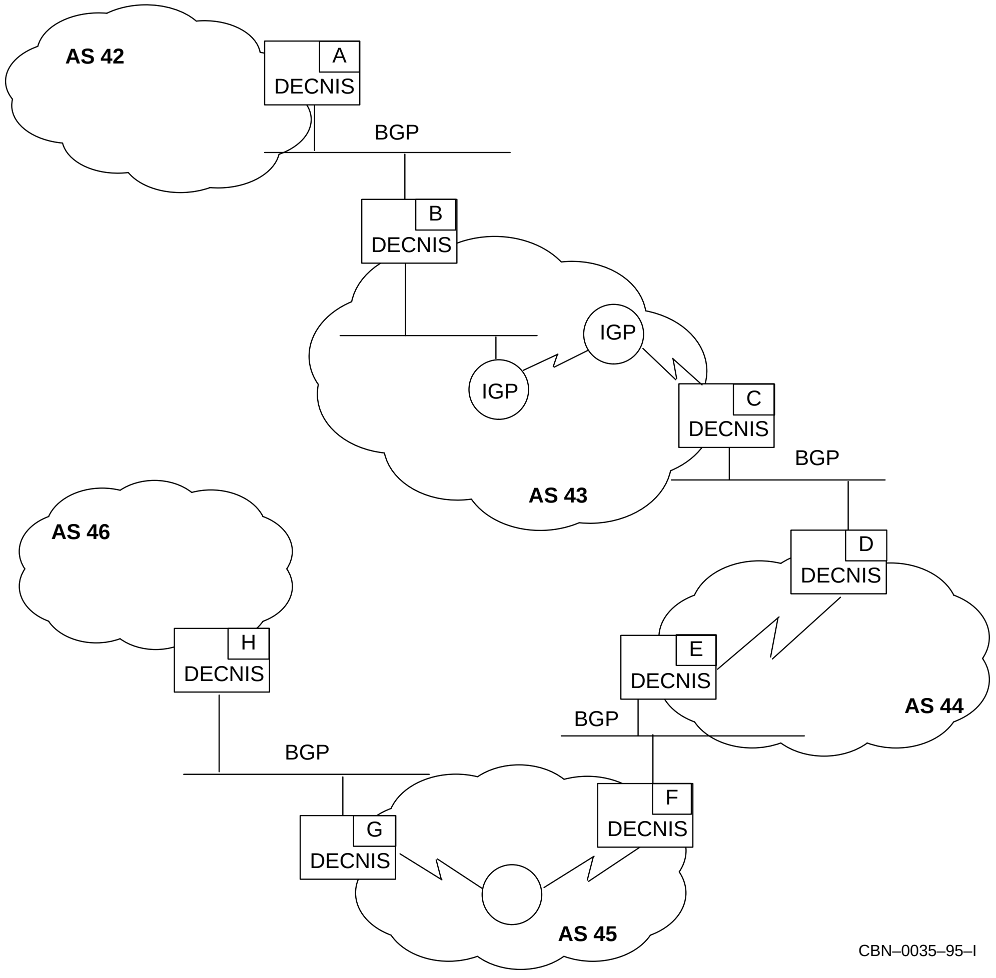
- They do not receive or propagate any routes from AS 45 or AS 46.
- They always receive and propagate routes originating in AS 44.
- They always receive and propagate routes learnt from AS 43.

General Method

You need to configure routers B and C as follows:

To do this...	Do this...
Block routes from AS 45 or AS 46	Set up a RECEIVE DESTINATION filter to exclude autonomous systems 45 and 46
Propagate routes originating in AS 44	Set up a PROPAGATE DESTINATION filter and specify ORIGINATING AUTONOMOUS SYSTEM as 44
Propagate routes learnt from AS 43	Set up a PROPAGATE DESTINATION filter and specify NEIGHBOR AUTONOMOUS SYSTEM as 43

Figure 12–7 Filtering Autonomous Systems



Procedure

Follow these steps on both DECNIS B and DECNIS C:

1. Create a RECEIVE DESTINATION filter for BGP, and specify the filter action as PASS.

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL bgp -
   _NCL> RECEIVE DESTINATION exclude FILTER ACTION PASS
   ```

2. Specify the address match type as PREFIX:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp -
   _NCL> RECEIVE DESTINATION exclude ADDRESS MATCH TYPE PREFIX
   ```

3. Exclude autonomous systems 45 and 46:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp RECEIVE -
   _NCL> DESTINATION exclude EXCLUDED AUTONOMOUS SYSTEMS {45, 46 }
   ```

   Any routes that include autonomous systems 45 or 46 in their AS path will not match this filter. All other routes will be passed.

4. Create a PROPAGATE DESTINATION filter for BGP, and specify the filter action as PASS:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL bgp PROPAGATE -
   _NCL> DESTINATION filter_PROPAGATE FILTER ACTION PASS
   ```

5. Specify the address match type as PREFIX:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp PROPAGATE -
   _NCL> DESTINATION filter_origin ADDRESS MATCH TYPE PREFIX
   ```

6. Specify that all routes originating in AS 44 should be propagated:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp -
   _NCL> PROPAGATE DESTINATION filter_origin -
   _NCL> ORIGINATING AUTONOMOUS SYSTEM 44 -
   ```

7. Create another PROPAGATE DESTINATION filter for BGP, and specify the filter action as PASS:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL bgp -
   _NCL> PROPAGATE DESTINATION filter_nbr FILTER ACTION PASS
   ```

8. Specify the address match type as PREFIX:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL bgp -
   _NCL> PROPAGATE DESTINATION filter_nbr ADDRESS MATCH TYPE PREFIX
   ```

9. Specify that routes learnt from AS 43 should be propagated:

```
NCL> SET ROUTING CONTROL PROTOCOL bgp -
_NCL> PROPAGATE DESTINATION filter_nbr -
_NCL> NEIGHBOR AUTONOMOUS SYSTEM 43
```

10. Enable the entities:

```
NCL> ENABLE  ROUTING CONTROL PROTOCOL bgp -
_NCL> RECEIVE DESTINATION exclude
NCL> ENABLE  ROUTING CONTROL PROTOCOL bgp -
_NCL> PROPAGATE DESTINATION filter_origin
NCL> ENABLE  ROUTING CONTROL PROTOCOL bgp -
_NCL> PROPAGATE DESTINATION filter_nbr
NCL> ENABLE  ROUTING CONTROL PROTOCOL bgp
```

13

IP Packet Filtering

13.1 Introduction

This chapter describes how to use IP packet filtering on the DECNIS.

IP packet filtering allows you to control the access between IP hosts on your network. This allows you to provide security on the IP section of your network. It also gives you some control over the bandwidth used on your network.

13.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5.

13.2 How IP Packet Filtering Works

You set up filters on the DECNIS which describe the type of IP packets you want the DECNIS to pass (transmit) and those you want it to block.

The DECNIS checks each IP packet it receives against the filters you have set up and determines whether to pass or block the packet.

Note the following:

- The DECNIS can only filter IP packets. However, for other types of packet you can use packet prioritization to provide a form of packet filtering (see Chapter 7).
- The DECNIS only filters packets on its outgoing interfaces; you cannot use IP packet filtering to prevent the DECNIS receiving IP packets.
- You cannot use IP filtering to filter X.25 packets.

Throughout this chapter, whenever the term packet filtering is used, it refers to IP packet filtering.

13.3 Example Uses for IP Packet Filtering

Figure 13–1 shows an example IP network of a large company. In this network you might want to control the internal IP access between different departments and the external IP access to customers over the Internet.

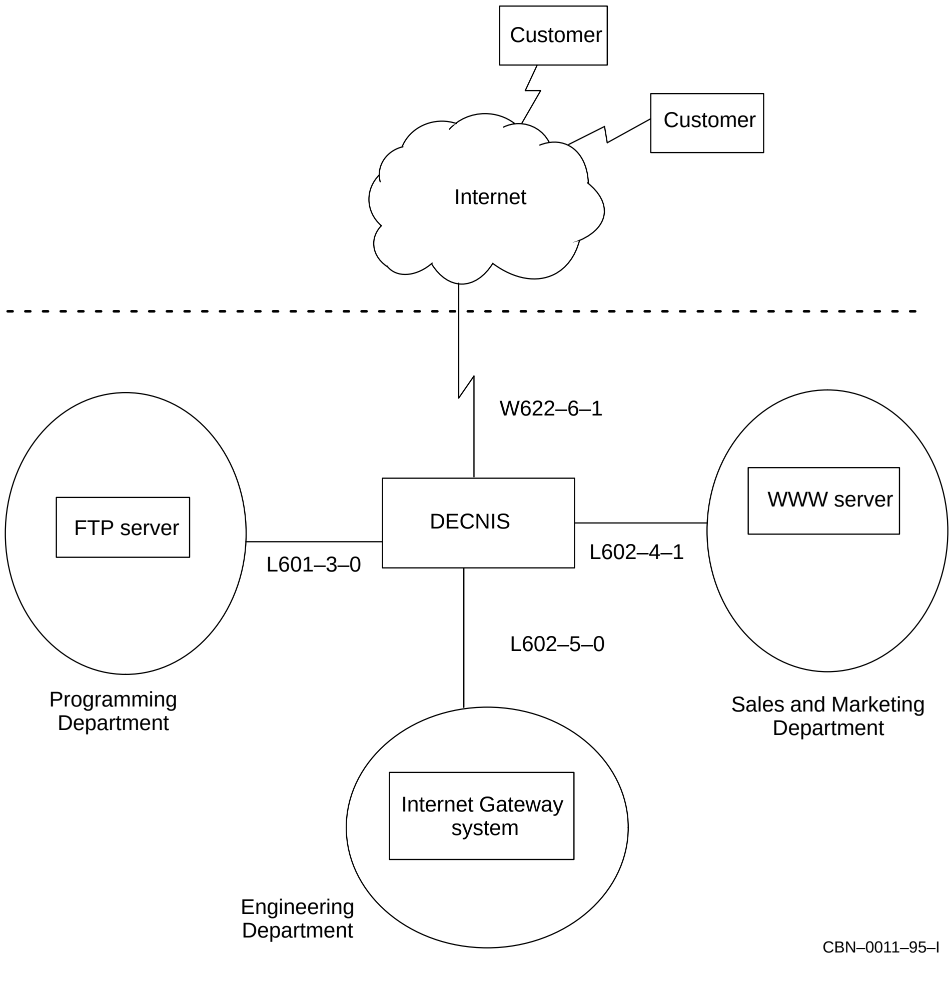
For example, in Figure 13–1, you might use packet filters to allow only the following types of IP access:

- Allow access to the FTP server from the Internet. This would allow files and images to be copied between the company and its customers.
- Allow access to the WWW (World-Wide Web) server from the Internet and all internal departments.
- Allow access between the Internet and the Internet Gateway
- Allow Telnet access from the Programming department to the Engineering department.

By default the DECNIS will prevent other types of IP access on any interface which has packet filtering activated (see Section 13.4.2). However, the DECNIS will still allow non-IP access such as DECnet/OSI.

Section 13.11 shows how to set up the IP packet filtering in this example.

Figure 13–1 Example IP Network



13.4 The PRIORITY Entities That Control IP Packet Filtering

Packet filtering is controlled by the PRIORITY FILTER and PRIORITY INTERFACE FILTER entities. These are subentities of the PRIORITY module, which also controls packet prioritization (see Chapter 7).

The DECNIS filters packets on its outgoing interfaces. For each interface on which you require packets to be filtered, you create a PRIORITY INTERFACE entity.

You then create either a global filter which applies to all interfaces or a single filter for each interface.

For a global filter, you create a PRIORITY FILTER entity. For a single interface filter, you create a PRIORITY INTERFACE FILTER entity.

For example, in Figure 13–1, if you wanted to allow Telnet packets to all departments, you would set up a global FILTER entity. If you wanted to allow only Telnet packets destined for the Engineering department, you would set up an INTERFACE FILTER entity for the L602-5-0 interface.

13.4.1 Maximum Number of Filters

Each PRIORITY INTERFACE entity (except those associated with the W618 and W614 Network Interface Cards) can support 94 filters. These can be made up of both global and interface filters.

The W618 and W614 cards can only support a total of 94 filters shared between all of their associated PRIORITY INTERFACE entities. For example, a W614 card could have 23 filters on three of its PRIORITY INTERFACE entities and 25 filters on its fourth (23 on W614-5-0, 23 on W614-5-1, 23 on W614-5-2 and 25 on W614-5-3).

13.4.2 Activating Packet Filtering

If packet filtering is not activated, by default the DECNIS passes all IP packets.

If packet filtering is activated, by default the DECNIS blocks all IP packets that do not match a PASS filter.

Packet filtering is activated on all interfaces when the PRIORITY module has been enabled and at least one global FILTER entity has been created.

Packet filtering is activated on a single interface when the PRIORITY module has been enabled and at least one INTERFACE FILTER entity has been created for the interface.

If packet filtering is activated on only one or two interfaces, by default the DECNIS blocks all IP packets on those interfaces, but passes IP packets on all other interfaces.

If packet filtering is activated and packet prioritization is set up, then the prioritization settings will only apply to the packets that match a PASS filter.

13.5 Types of Filter

There are different types of filter for different protocol types.

When you create a PRIORITY FILTER entity or a PRIORITY INTERFACE FILTER entity, you must set the TYPE attribute of the entity. This identifies the protocol type of the packets that it will filter, and whether it will Block or Pass the packets. There are eight types of filter:

- Block TCP
- Pass TCP
- Block UDP
- Pass UDP
- Block ICMP
- Pass ICMP
- Block IP Protocol
- Pass IP Protocol

A filter with a TYPE attribute of Block IP Protocol or Pass IP Protocol will filter all IP protocols except TCP, UDP or ICMP.

13.6 Packet Characteristics

This section describes the characteristics that the DECNIS uses to filter different types of IP packet.

13.6.1 All IP Packets

The DECNIS can filter all types of IP packet according to the following characteristics:

- The Inbound Interface on which the packet was received.

  For example, all packets received on L601-3-0.

- The Source Address of the packet.

  For example, all packets with the source address 16.32.24.1 (using the mask 255.255.255.255).

- The Destination Address of the packet.

  For example, all packets with the destination subnet 37.17.11.0 (using the mask 255.255.255.0).

13.6.2 TCP and UDP Packets

TCP and UDP packets can be filtered according to their Destination Port number or Source Port number. This allows you to filter different services and protocols. For example, to filter Telnet packets, you would specify 23 for the destination port number (using the mask 65535).

Note that for Port number characteristics in TCP you normally need two filters; one for source and one for destination for the reverse traffic.

Table 7–2 lists some commonly used services and protocols and their assigned TCP or UDP port numbers. For a complete listing of port numbers, refer to RFC 1700 (October 1994).

13.6.3 ICMP Packets

ICMP packets can be filtered according to their ICMP Message Type. For example, to filter ECHO packets used in PING, you would specify an ICMP message type of 8 (using the mask 65535).

13.6.4 Non TCP, UDP or ICMP Packets

Non TCP, UDP or ICMP packets can be filtered according to their IP Protocol number. For example, you could filter EGP packets by specifying the IP protocol number 8 (using the mask 65535).

Table 7–1 lists some commonly used protocols and their assigned protocol numbers. For a complete listing of protocol numbers, refer to RFC 1700 (October 1994).

13.6.5 Combinations of Characteristics

You can use one characteristic or a combination of characteristics to describe the packet you want to filter. For example, by specifying 23 for the destination port number, you can filter Telnet packets.

By also specifying the destination address and inbound interface characteristics, you can be more specific about which Telnet packets you want to filter.

Note: To make your IP filtering set up more secure you should always specify the Inbound Interface characteristic whenever you use the Source Address or Source Port characteristics.

This is because it is possible for an unauthorized system to use packets with a false source address or source port.

13.7 Filtering Order for IP Packets

Figure 13–2 shows what happens when packet filtering is activated (see Section 13.4.2) and the DECNIS performs filtering on an IP packet it is about to transmit.

First the DECNIS looks at the protocol type and the characteristics of the packet.

The DECNIS then looks for a matching filter with the same protocol type and the same characteristics as the packet. The DECNIS always follows the same order of checking:

1. Block INTERFACE FILTER
2. Pass INTERFACE FILTER
3. Block FILTER
4. Pass FILTER

If the DECNIS finds a matching filter, it will block or pass (transmit) the packet according to the TYPE attribute of the filter.

If the DECNIS cannot find any matching filters, it will block the packet.

This is a security measure to ensure that once you have activated packet filtering, only packets which match a pass filter will be transmitted.

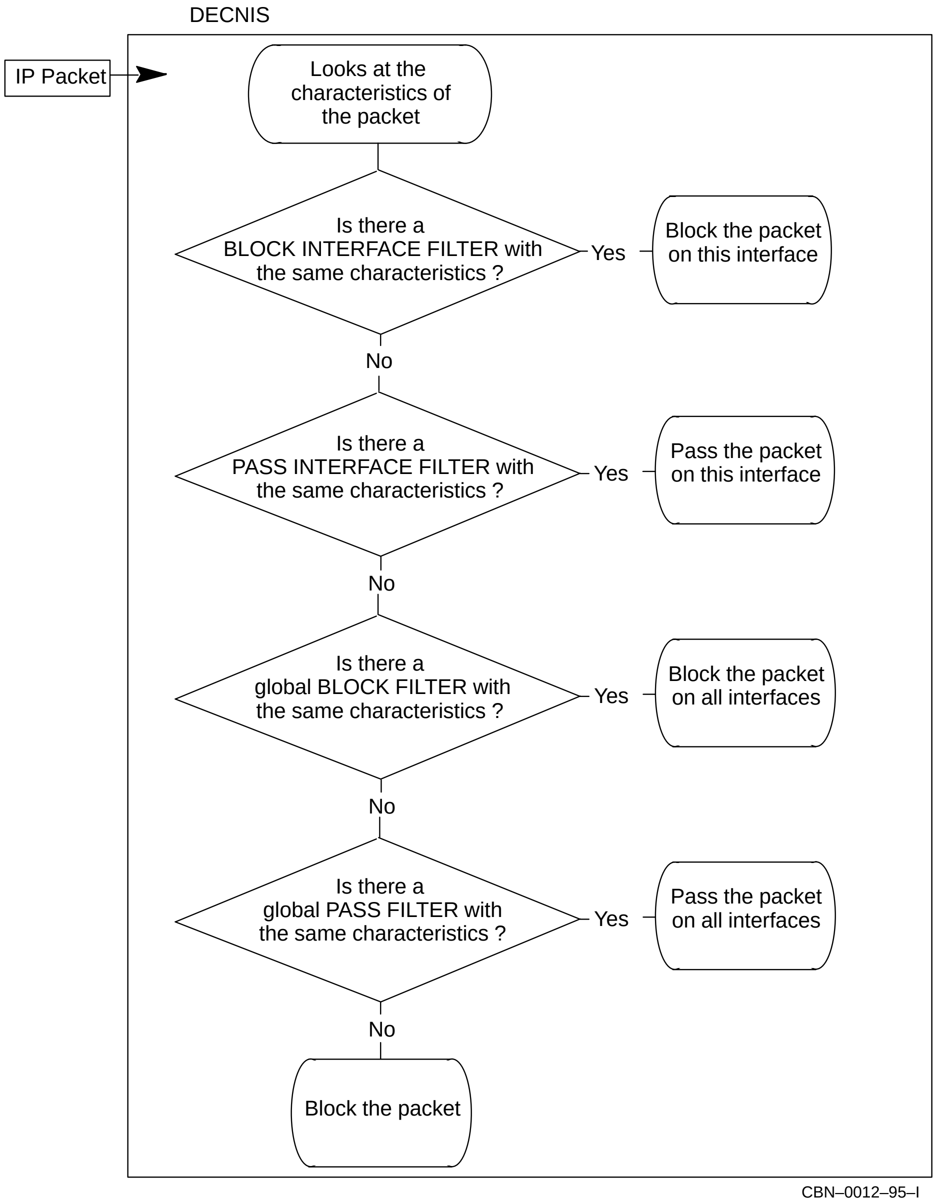
This strict order of checking means that Interface filters always take precedence over Global filters, and Block type filters always take precedence over Pass type filters.

It is important to remember this order of checking when you are setting up packet filtering. For example, if you create a global FILTER to block all Telnet packets but there is already an INTERFACE FILTER to pass Telnet packets, then Telnet will always be passed on that interface.

13.7.1 Masks and Matching

Where filters are specified using values and masks, a mask of 0 matches everything. A mask of 255.255.255.255 or 65535 requires an exact match. Parts of the value field which are not in the mask are ignored.

Figure 13–2 Filtering Order for IP Packets


DECNIS
IP Packet
Looks at the characteristics of the packet
Is there a BLOCK INTERFACE FILTER with the same characteristics ?
Yes
Block the packet on this interface
No
Is there a PASS INTERFACE FILTER with the same characteristics ?
Yes
Pass the packet on this interface
No
Is there a global BLOCK FILTER with the same characteristics ?
Yes
Block the packet on all interfaces
No
Is there a global PASS FILTER with the same characteristics ?
Yes
Pass the packet on all interfaces
No
Block the packet
CBN–0012–95–I

13.8 How to Set Up Packet Filtering For IP

To set up packet filtering on the DECNIS, you need to carry out the following tasks:

1. Plan your packet filtering requirements.
2. Create or Uncouple the PRIORITY module.
3. Create and enable a PRIORITY INTERFACE entity for every outgoing interface you want to filter packets on, and every inbound interface you want to specify as an Inbound Interface characteristic in a filter.
4. Set up a PRIORITY INTERFACE FILTER entity for each type of packet that you want to filter on one specific outgoing interface.
5. Set up a global PRIORITY FILTER entity for each type of packet you want to filter on all outgoing interfaces.
6. Enable or Update the PRIORITY module.

These tasks are described in Section 13.8.1 to Section 13.8.6.

13.8.1 Plan Your Packet Filtering Requirements

The most secure way to set up packet filtering is to set up filters to pass the packets you want to allow. By default the DECNIS will block all other packets.

This means you need to decide on the characteristics of all the packets you want to allow.

If you forget to set up filters for some packets and you find that they are being blocked, it is easy to set up new filters to pass the packets.

The other approach to packet filtering is to set up filters to block unwanted packets. However you must then set up "pass-all" filters to pass all other types of packet. This method may not be as secure as the first method. For example, if you forget to block a particular protocol type, the packet may be forwarded in a pass-all filter. This might cause a security problem.

To plan your packet filtering requirements, you need to:

- Decide which method of filtering to use:

  Pass specific packets and use the default block

  Block specific packets and use "Pass-all" filters

- Map out your IP network and determine which interfaces connect to which subnets and hosts. You then need to decide whether you need interface specific filters, global filters or both.

- Gather all the information you need to create your filters. Use the template in Table 13–1 to help you. The first column in Table 13–1 is an example of the information required to create a filter to pass Telnet packets on the outbound interface l601-3-0, from the source address 16.32.16.1, and received on the inbound interface w622-4-0.

  Section 13.11.1 shows example NCL commands required to set up the example packet filtering situation in Figure 13–3.

Table 13–1 Template for Setting Up Packet Filters

	Example			
Characteristics	**Filter 1**	**Filter 2**	**Filter 3**	**Filter 4**
Interface Filter or Global Filter	interface l601-3-0			
Filter Name	pass_telnet			
Type	pass tcp			
Inbound Interface	w622-4-0			
Source Address Value	16.32.16.1			
Source Address Mask	255.255.255.255			
Destination Address Value	–			
Destination Address Mask	0.0.0.0			
Source Port Value	–			
Source Port Mask	0			
Destination Port Value	23			
Destination Port Mask	65535			
ICMP Message Type	–			
ICMP Message Mask	–			
IP Protocol Value	–			
IP Protocol Mask	–			

13.8.2 Create or Uncouple the PRIORITY Module

To set up packet filtering the PRIORITY module needs to exist and its Configuration State must be set to Off or Uncoupled.

Enter the following command:

```
NCL> SHOW PRIORITY CONFIGURATION STATE
```

If the PRIORITY module does not exist, you will receive the following error message:

```
command failed due to:
 no such object instance
```

To create the PRIORITY module, enter the following command:

```
NCL> CREATE PRIORITY
```

If the PRIORITY module exists but its Configuration state is Updated, you must Uncouple the PRIORITY module.

The Configuration State of Updated indicates that the PRIORITY module is already being used for either packet prioritization or existing IP packet filters. In this situation, the Uncouple command allows you to create new packet filters without affecting the current operation of any existing packet filters or prioritization settings.

Once you have created your packet filters you then issue the Update command. This will replace any existing packet filters with the new filters you have created but will not affect your prioritization settings.

To Uncouple the PRIORITY module, enter the following command:

```
NCL> UNCOUPLE PRIORITY
```

The Update command is described in Section 13.8.6.

13.8.3 Create and Enable PRIORITY INTERFACE Entities

You need to create and enable a PRIORITY INTERFACE entity for:

- Every outgoing interface you want to filter packets on.
- Every interface which will be specified as an Inbound Interface characteristic in a filter.

Note: If you are already using packet filtering or packet prioritization, some PRIORITY INTERFACE entities will already exist.

To create and enable a PRIORITY INTERFACE entity, enter the following commands:

```
NCL> CREATE PRIORITY INTERFACE interface-name COMMUNICATIONS PORT com-port
NCL> ENABLE PRIORITY INTERFACE interface-name
```

where,

interface-name is the name of the interface.

com-port is the name of the hardware communication port associated with the interface. For information about the format of communication port names, refer to Section 8.3.

It is advisable to give the INTERFACE entity the same name as the communications port or the same name as the routing circuit.

Example

To create and enable an INTERFACE entity for the L601-3-0 interface, enter the following command:

```
NCL> CREATE PRIORITY INTERFACE l601-3-0 COMMUNICATIONS PORT l601-3-0
NCL> ENABLE PRIORITY INTERFACE l601-3-0
```

13.8.4 Set Up Interface Specific Filters

You need to set up an INTERFACE FILTER entity for every type of packet you want to filter on a specific interface.

To set up an INTERFACE FILTER entity, enter the following commands:

```
NCL> CREATE PRIORITY INTERFACE interface-name FILTER filter-name TYPE type
```

```
NCL> SET PRIORITY INTERFACE interface-name FILTER filter-name -
_NCL> characteristic chara-value, characteristic chara-value, ...
```

```
NCL> ENABLE PRIORITY INTERFACE interface-name FILTER filter-name
```

where,

interface-name	is the name of the interface to which the filter applies.
filter-name	is the name of the filter.
type	is the type of filter. This should be one of the Filter Types in Table 13–2.
characteristic	is one of the characteristics in Table 13–3.
chara-value	is one of the chara-values related to the characteristic in Table 13–3.

If you do not set all the possible characteristics for a packet, the DECNIS uses the default values and masks. The default values and masks for all characteristics are set to all zeros. This means that any value will match the characteristic.

For example, if you do not set the Destination Address Value and mask, the filter will match IP packets with any destination address.

Example

This example creates a filter which will pass Telnet packets (Destination Port 23) destined for the outbound interface l601-3-0, coming from the source address 16.32.16.1 and received on the inbound interface w622-4-0.

The packet can have any destination address and source port.

```
NCL> CREATE PRIORITY INTERFACE l601-3-0 FILTER pass_telnet TYPE pass tcp
```

```
NCL> SET PRIORITY INTERFACE l601-3-0 FILTER pass_telnet destination port -
_NCL> value 23, destination port mask 65535, inbound interface priority -
_NCL> interface w622-4-0, source address value 16.32.16.1, source address -
_NCL> mask 255.255.255.255
```

```
NCL> ENABLE PRIORITY INTERFACE l601-3-0 FILTER pass_telnet
```

The example filter "*pass_telnet*" uses the Destination Port 23 to match Telnet packets. For Telnet to work correctly, you also need a reverse filter to pass the Telnet responses (Source Port 23) back to the source. To create a reverse filter to pass Telnet responses back to the source address 16.32.16.1 on the w622-4-0 interface, enter the following commands:

```
NCL> CREATE PRIORITY INTERFACE w622-4-0 FILTER pass_telnet_src TYPE pass tcp
```

```
NCL> SET PRIORITY INTERFACE w622-4-0 FILTER pass_telnet_src -
_NCL> source port value 23, source port mask 65535, inbound -
_NCL> interface priority interface l601-3-0, destination address value -
_NCL> 16.32.16.1, destination address mask 255.255.255.255
```

```
NCL> ENABLE PRIORITY INTERFACE w622-4-0 FILTER pass_telnet_src
```

The example in Section 13.11 shows how pairs of filters (Source filter and Destination filter) are required when you filter using TCP Port numbers.

Note: It is possible to set characteristics on the CREATE command line, for example:

```
NCL> CREATE PRIORITY INTERFACE l601-3-0 FILTER pass_telnet TYPE pass tcp, -
_NCL> destination port value 23, destination port mask 65535, inbound -
_NCL> interface priority interface w622-4-0, source address value -
_NCL> 16.32.16.1, source address mask 255.255.255.255
```

Table 13–2 Filter Types

Protocol Type	Filter Type
TCP Packets	block tcp pass tcp
UDP Packets	block udp pass udp
ICMP Packets	block icmp pass icmp
Non-TCP, UDP or ICMP Packets	block ip protocol pass ip protocol

Table 13–3 IP Packet Filter Characteristics

Characteristic	chara-value	Example Value	Default Value
All Packets			
INBOUND INTERFACE	priority interface interface-name	priority interface w622-4-0	–
SOURCE ADDRESS VALUE	n.n.n.n	16.32.16.0	0.0.0.0
SOURCE ADDRESS MASK	n.n.n.n	255.255.255.0	0.0.0.0
DESTINATION ADDRESS VALUE	n.n.n.n	20.14.0.0	0.0.0.0
DESTINATION ADDRESS MASK	n.n.n.n	255.255.0.0	0.0.0.0
TCP and UDP Packets			
SOURCE PORT VALUE	n	23	0
SOURCE PORT MASK	n	65535	0
DESTINATION PORT VALUE	n	23	0
DESTINATION PORT MASK	n	65535	0
ICMP Packets			
ICMP MESSAGE TYPE	n	2	0
ICMP MESSAGE MASK	n	65535	0
Non TCP, UDP or ICMP Packets			
IP PROTOCOL VALUE	n	6	0
IP PROTOCOL MASK	n	255	255

13.8.5 Set Up Global Filters

You need to set up a global FILTER entity for every type of packet you want to filter on all interfaces.

To set up a FILTER entity, enter the following commands:

```
NCL> CREATE PRIORITY FILTER filter-name TYPE type
```

```
NCL> SET PRIORITY FILTER filter-name characteristic chara-value, -
_NCL> characteristic chara-value, ...
```

```
NCL> ENABLE PRIORITY FILTER filter-name
```

where,

filter-name	is the name of the filter.
type	is the type of filter. This should be one of the Filter Types in Table 13–2.
characteristic	is one of the characteristics in Table 13–3.
chara-value	is one of the chara-values related to the characteristic in Table 13–3.

If you do not set all the possible characteristics for a packet, the DECNIS uses the default values and masks. The default values and masks for all characteristics are set to all zeros. This means that any value will match the characteristic.

For example, if you do not set the Destination Address Value and mask, the filter will match IP packets with any destination address.

Example

This example creates a filter which will pass Telnet packets (Destination Port 23) destined for any outbound interface, coming from the source address 16.32.16.1, and received on the inbound interface w622-4-0.

The packets can have any destination address and source port.

```
NCL> CREATE PRIORITY FILTER pass_telnet_global TYPE pass tcp
```

```
NCL> SET PRIORITY FILTER pass_telnet_global destination port value 23, -
_NCL> destination port mask 65535, inbound interface priority interface -
_NCL> w622-4-0, source address value 16.32.16.1, source address -
_NCL> mask 255.255.255.255
```

```
NCL> ENABLE PRIORITY FILTER pass_telnet_global
```

The example filter "*pass_telnet_global*" uses the Destination Port 23 to match Telnet packets. For Telnet to work correctly, you also need to create a reverse filter to pass the Telnet responses (Source Port 23) back to the source. To create a reverse filter to pass Telnet responses back to the source address 16.32.16.1 on from any inbound interface, enter the following commands:

```
NCL> CREATE PRIORITY FILTER pass_telnet_global_src TYPE pass tcp
```

```
NCL> SET PRIORITY FILTER pass_telnet_global_src source -
_NCL> port value 23, source port mask 65535, destination address value -
_NCL> 16.32.16.1, destination address mask 255.255.255.255
```

```
NCL> ENABLE PRIORITY FILTER pass_telnet_global_src TYPE pass tcp
```

The example in Section 13.11 shows how pairs of filters (source filter and destination filter) are required when you filter using TCP Port numbers.

Note: It is possible to set characteristics on the CREATE command line, for example:

```
NCL> CREATE PRIORITY FILTER pass_telnet_global TYPE pass tcp, destination -
_NCL> port value 23, destination port mask 65535, inbound interface -
_NCL> priority interface w622-4-0, source address value 16.32.16.1, -
_NCL> source address mask 255.255.255.255
```

13.8.6 Enable or Update the PRIORITY Module

If you created the PRIORITY module in Section 13.8.2, you now need to enable it by entering the following command:

```
NCL> ENABLE PRIORITY
```

If the PRIORITY module was already created and you had to use the Uncouple command in Section 13.8.2, you now need to Update the PRIORITY module by entering the following command:

```
NCL> UPDATE PRIORITY
```

The Update command makes your new packet filters apply to the DECNIS. If there were already packet filters in operation, these are now replaced.

13.9 Modifying Your Packet Filtering Set Up

If you need to change your packet filtering set up, for example, to add or delete filters, you must Uncouple the PRIORITY module.

The Uncouple command allows you to modify your packet filtering set up without affecting the current operation of any existing packet filters or any prioritization settings.

Once you have created, modified or deleted your packet filters you then issue the Update command. This will replace the existing packet filtering set up with the new packet filtering set up.

To Uncouple the PRIORITY module, enter the following command:

```
NCL> UNCOUPLE PRIORITY
```

To Update the PRIORITY module, enter the following command:

```
NCL> UPDATE PRIORITY
```

13.10 Packet Filtering Restrictions

13.10.1 Using SNMP to Manage the DECNIS

If you want to use SNMP to manage a DECNIS over an interface which has packet filtering activated, you must set up a filter to pass SNMP responses on that interface.

Because filtering only takes place on the outgoing circuit, you cannot prevent the DECNIS from receiving and processing SNMP requests. However the DECNIS will not send an acknowledgement of the SNMP request unless you set up a filter to pass SNMP responses.

This can give a false impression of the actions of the DECNIS. For example, if the DECNIS is sent an SNMP SET request, it will process the SET but will not send an acknowledgement that the SET has taken place.

If you forget to set up a filter to pass SNMP responses and packet filtering is activated on the interface you are trying to use, you will not be able to use SNMP to manage the DECNIS on that interface.

If you are unable to use SNMP on any of the DECNIS interfaces, you could use a DECnet/OSI connection to Uncouple packet filtering and set up an SNMP filter. The other option would be to reboot the DECNIS with a new packet filtering NCL script (containing the SNMP filter).

Note: You cannot specify the DECNIS system's MPC as an Inbound Interface characteristic. If you want to pass only SNMP packets from the DECNIS system's MPC, you should use the IP address of the DECNIS as the Source Address characteristic.

13.10.2 Cannot Stop the DECNIS Receiving Packets

It is important to remember that packet filtering only takes place on the outgoing circuit and can therefore only prevent the transmission of packets. This means that you cannot use packets filters to prevent the DECNIS from receiving and processing IP packets.

For example, you cannot prevent the DECNIS from receiving and processing SNMP SET requests. If packet filtering is activated, the DECNIS will receive the SNMP SET request and process it. However it will not send an acknowledgement that the SET has taken place unless you set up a filter to pass SNMP responses (see Section 13.10.1).

13.10.3 Fragmented Packets

Packet filtering will not work as expected if packets are fragmented. If a packet is fragmented it is likely that only the first fragment will match a packet filter. The remaining fragments of the packet will be blocked.

It may be possible to work around this problem by using a filter of the type Pass IP Protocol to pass the remaining fragments.

13.10.4 Filtering FTP Packets

Some FTP applications use non-default data port numbers in their packets. This makes it impossible to set up filters to pass the FTP packets because you do not know what the source port will be at any one time.

Any FTP examples in this chapter assume that default data port number 20 is used. For more information about FTP data ports and packet filtering, refer to RFC 1579.

13.11 Setting Up an Example Packet Filtering Situation

This section describes how to set up packet filters in the example IP network shown in Figure 13–3. (This is the same as the example used in Section 13.3.)

The templates in Table 13–4, Table 13–5 and Table 13–6 contain the characteristics that need to be set up in each filter.

Section 13.11.1 describes the NCL commands required to set up each packet filter.

In this example packet filtering allows only the following types of access:

- Allow access to the FTP server from the Internet. (Assumes that default data port (20) is used.)
- Allow access to the WWW (World-Wide Web) server from the Internet and all internal departments.
- Allow access between the Internet and the Internet Gateway.
- Allow Telnet access from the Programming department to the Engineering department.
- Allow SNMP responses from the DECNIS to the Engineering department. (Allows SNMP management of the DECNIS.)

Figure 13–3 Example IP Network

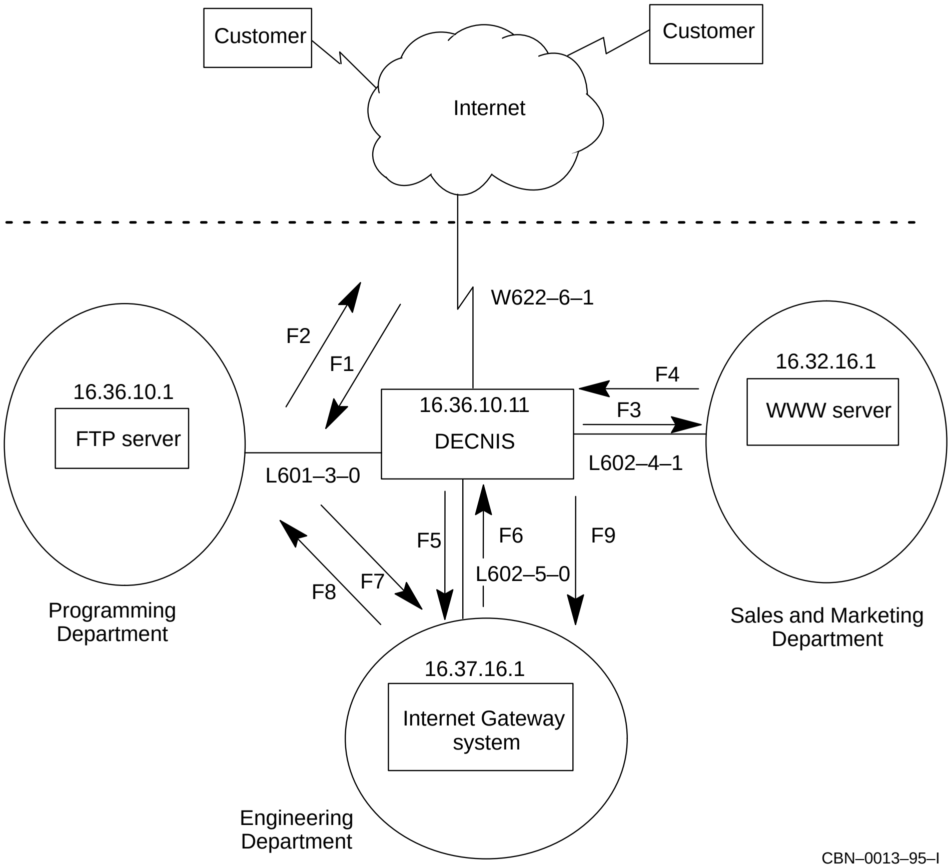


Table 13–4 Filters Required For the Example IP Network

Characteristics	Filter 1 (F1)	Filter 2 (F2)	Filter 3 (F3)
Interface Filter or Global Filter	interface l601-3-0	interface w622-6-1	interface l602-4-1
Filter Name	pass_ftp_Inet	pass_ftp_Serv	pass_www_Inet
Type	pass tcp	pass tcp	pass tcp
Inbound Interface	w622-6-1	l601-3-0	–
Source Address Value	–	16.36.10.1	–
Source Address Mask	–	255.255.255.255	–
Destination Address Value	16.36.10.1	–	16.32.16.1
Destination Address Mask	255.255.255.255	–	255.255.255.255
Source Port Value	–	20	–
Source Port Mask	–	65534[1]	–
Destination Port Value	20	–	80
Destination Port Mask	65534[1]	–	65535
ICMP Message Type	–	–	–
ICMP Message Mask	–	–	–
IP Protocol Value	–	–	–
IP Protocol Mask	–	–	–

[1] A port mask of 65534 with the port value of 20, matches port values 20 and 21.

Table 13–5 Filters Required For the Example IP Network Continued..

Characteristics	Filter 4 (F4)	Filter 5 (F5)	Filter 6 (F6)
Interface Filter or Global Filter	global	interface l602-5-0	interface w622-6-1
Filter Name	pass_www_glbl	pass_from_Inet	pass_from_ Gway
Type	pass tcp	pass tcp	pass tcp
Inbound Interface	l602-4-1	w622-6-1	l602-5-0
Source Address Value	16.32.16.1	–	16.37.16.1
Source Address Mask	255.255.255.255	–	255.255.255.255
Destination Address Value	–	16.37.16.1	–
Destination Address Mask	–	255.255.255.255	–
Source Port Value	80	–	–
Source Port Mask	65535	–	–
Destination Port Value	–	–	–
Destination Port Mask	–	–	–
ICMP Message Type	–	–	–
ICMP Message Mask	–	–	–
IP Protocol Value	–	–	–
IP Protocol Mask	–	–	–

Table 13–6 Filters Required For the Example IP Network Continued..

Characteristics	Filter 7 (F7)	Filter 8 (F8)	Filter 9 (F9)
Interface Filter or Global Filter	interface l602-5-0	interface l601-3-0	interface l602-5-0
Filter Name	pass_Tnet_dest	pass_Tnet_src	pass_snmp
Type	pass tcp	pass tcp	pass udp
Inbound Interface	l601-3-0	l602-5-0	–
Source Address Value	–	16.37.16.1	16.36.10.11
Source Address Mask	–	255.255.255.255	255.255.255.255
Destination Address Value	16.37.16.1	–	16.37.0.0
Destination Address Mask	255.255.255.255	–	255.255.0.0
Source Port Value	–	23	161
Source Port Mask	–	65535	65535
Destination Port Value	23	–	–
Destination Port Mask	65535	–	–
ICMP Message Type	–	–	–
ICMP Message Mask	–	–	–
IP Protocol Value	–	–	–
IP Protocol Mask	–	–	–

13.11.1 Example NCL

The following NCL script contains the commands required to set up the packet filters in Figure 13–3. These NCL commands are included in an example script file. From the example script file, you can copy the NCL commands you require into the user NCL script files for your DECNIS. For more details about the user NCL script files, refer to Section 1.5.6.

For details about the location of all the example script files, refer to Appendix G.

```
NCL> create priority

NCL> create priority interface l601-3-0 communication port l601-3-0
NCL> enable priority interface l601-3-0

NCL> create priority interface w622-6-1 communication port w622-6-1
NCL> enable priority interface w622-6-1

NCL> create priority interface l602-4-1 communication port l602-4-1
NCL> enable priority interface l602-4-1
NCL> create priority interface l602-5-0 communication port l602-5-0
NCL> enable priority interface l602-5-0


NCL> create priority interface l601-3-0 filter pass_ftp_Inet type pass tcp

NCL> set priority interface l601-3-0 filter pass_ftp_Inet -
_NCL> inbound interface priority interface w622-6-1, -
_NCL> destination address value 16.36.10.1, destination address -
_NCL> mask 255.255.255.255, destination port value 20, destination -
_NCL> port mask 65534

NCL> enable priority interface l601-3-0 filter pass_ftp_Inet


NCL> create priority interface w622-6-1 filter pass_ftp_Serv type pass tcp

NCL> set priority interface w622-6-1 filter pass_ftp_Serv -
_NCL> inbound interface priority interface l601-3-0, -
_NCL> source address value 16.36.10.1, source address -
_NCL> mask 255.255.255.255, source port value 20, source -
_NCL> port mask 65534

NCL> enable priority interface w622-6-1 filter pass_ftp_Serv


NCL> create priority interface l602-4-1 filter pass_www_Inet type pass tcp

NCL> set priority interface l602-4-1 filter pass_www_Inet -
_NCL> destination address value 16.32.16.1, destination address -
_NCL> mask 255.255.255.255, destination port value 80, destination -
_NCL> port mask 65535

NCL> enable priority interface l602-4-1 filter pass_www_Inet


NCL> create priority filter pass_www_glbl type pass tcp

NCL> set priority filter pass_www_glbl inbound interface priority -
_NCL> interface l602-4-1, source address value 16.32.16.1, source address -
_NCL> mask 255.255.255.255, source port value 80, source port mask 65535

NCL> enable priority filter pass_www_glbl


NCL> create priority interface l602-5-0 filter pass_from_Inet type pass tcp

NCL> set priority interface l602-5-0 filter pass_from_Inet -
_NCL> inbound interface priority interface w622-6-1, -
_NCL> destination address value 16.37.16.1, destination address -
_NCL> mask 255.255.255.255

NCL> enable priority interface l602-5-0 filter pass_from_Inet


NCL> create priority interface w622-6-1 filter pass_from_Gway type pass tcp

NCL> set priority interface w622-6-1 filter pass_from_Gway -
_NCL> inbound interface priority interface l602-5-0, -
_NCL> source address value 16.37.16.1, source address -
_NCL> mask 255.255.255.255
```

```
NCL> enable priority interface w622-6-1 filter pass_from_Gway

NCL> create priority interface l602-5-0 filter pass_Tnet_dest type pass tcp

NCL> set priority interface l602-5-0 filter pass_Tnet_dest -
_NCL> inbound interface priority interface l601-3-0, -
_NCL> destination address value 16.37.16.1, destination address -
_NCL> mask 255.255.255.255, destination port value 23, -
_NCL> destination port mask 65535

NCL> enable priority interface l602-5-0 filter pass_Tnet_dest

NCL> create priority interface l601-3-0 filter pass_Tnet_src type pass tcp

NCL> set priority interface l601-3-0 filter pass_Tnet_src -
_NCL> inbound interface priority interface l602-5-0, -
_NCL> source address value 16.37.16.1, source address -
_NCL> mask 255.255.255.255, source port value 23, source port mask 65535

NCL> enable priority interface l601-3-0 filter pass_Tnet_src

NCL> create priority interface l602-5-0 filter pass_snmp type pass udp

NCL> set priority interface l602-5-0 filter pass_snmp -
_NCL> destination address value 16.37.0.0, destination address -
_NCL> mask 255.255.0.0, source port value 161, source port mask 65535, -
_NCL> source address value 16.36.10.11, source address mask 255.255.255.255

NCL> enable priority interface l602-5-0 filter pass_snmp

NCL> enable priority
```

14
IP Multicasting

This chapter describes IP multicasting and how to set up the DECNIS as an IP multicast router.

14.1 IP Multicasting Definitions

The DECNIS supports IP multicasting as defined in:

- RFC 1112—Host Extensions For IP Multicasting
- Internet Draft—Protocol Independent Multicast (PIM) Protocol Specification, January 1995
- Internet Draft—Protocol Independent Multicast (PIM) Motivation and Architecture, January 1995

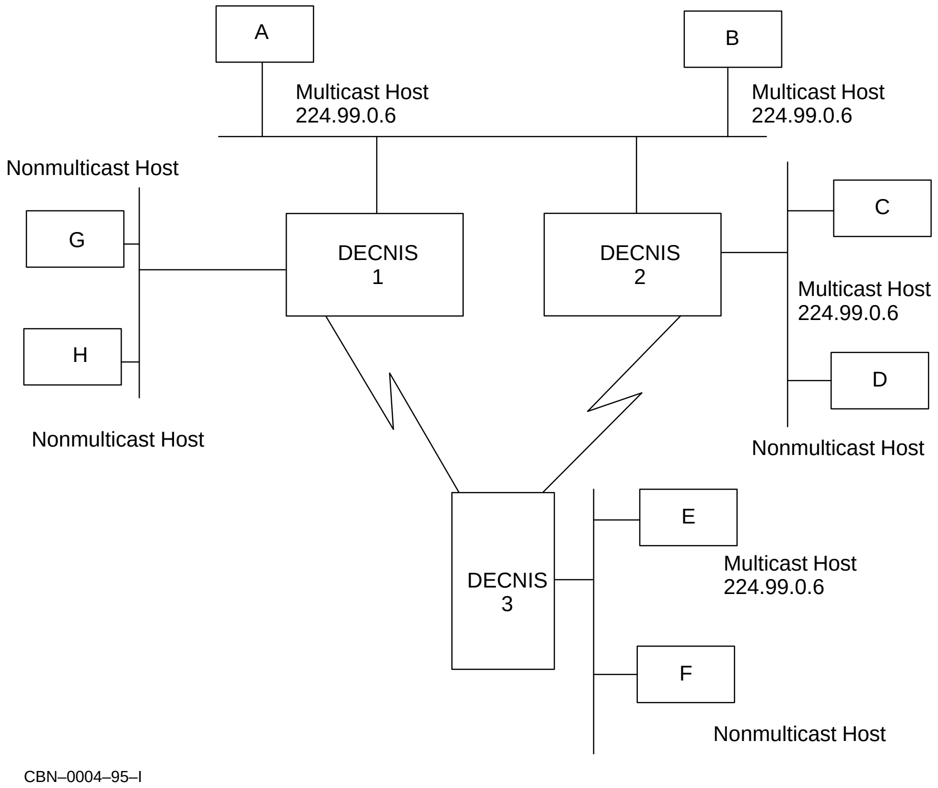
14.2 What Is IP Multicasting?

IP multicasting is a way of forwarding data to a group of host systems simultaneously. It is similar to IP broadcasting except that instead of all hosts receiving the data, only systems which belong to a multicast "host group" receive the data. This means that the overheads are reduced for hosts which are not members of the group.

A multicast host group is a group of systems which have the same Class D IP destination address. Class D IP addresses have 1110 as the high-order four bits: they therefore range from 224.0.0.0 to 239.255.255.255. Note that the addresses 224.0.0.0 to 224.0.0.255 are reserved by RFC 1700 and will not be forwarded by the DECNIS.

In Figure 14–1, the hosts A, B, C and E are all members of a host group with the Class D IP address 224.99.0.6.

Figure 14–1 DECNIS Systems Acting as IP Multicast Routers



There is no restriction on the location or numbers of hosts in the group. A host can join or leave a host group at any time and can be a member of more than one group.

IP multicast packets are delivered to all members of the host group with the same "best effort" reliability as unicast packets.

A host transmits an IP multicast packet over a LAN and it reaches all immediately neighboring members of the host group. Any multicast routers on the LAN forward the packet to any other networks that have members of the host group.

14.3 Example Uses For IP Multicast

IP multicasting is useful for sending out information to a large group of systems or requesting information from a large group of systems. For example, IP multicasting could be used by a "supervisor" type application. A "supervisor" system could monitor the status of all members of a host group by sending out regular status requests. All systems that are members of the host group at that time will receive the message and respond with a status report.

A "supervisor" system might also send out an update of available resources, for example, the location of a file server. A host could join or leave the host group depending on whether it required an update of the available resources.

The DECNIS implementation of IP multicasting provides good performance for applications such as the supervisor example. However, the DECNIS currently implements IP multicast on the MPC–II (Management Processor Card) and therefore it is not suitable for high throughput applications such as multicast video conferencing.

14.4 The Protocols That Control IP Multicasting

IP multicasting on the DECNIS is controlled by the following protocols:

- IGMP (Internet Group Management Protocol)

  This is used by multicast hosts to report their group memberships to any immediately neighboring multicast routers.

- PIM (Protocol Independent Multicast)

  This is used by multicast routers to determine which other multicast routers should receive multicast packets.

Figure 14–2 shows the relationship between the IGMP and PIM protocols and the hosts and routers.

14.4.1 PIM Protocol Modes

The PIM protocol can operate in two modes:

- Dense–Mode
- Sparse–Mode

Dense–Mode PIM transmits packets to all LANs unless it receives instructions to the contrary. This makes it suitable for networks with large numbers of multicast hosts.

Sparse–Mode PIM only transmits packets to LANs which have made "join" requests. This makes it suitable for networks with fewer or more widely distributed multicast hosts.

The DECNIS uses Dense–Mode PIM.

14.4.2 Dense-Mode PIM

The Dense–Mode PIM protocol uses reverse path forwarding. When a multicast router receives a multicast packet, it looks up the source of the packet in its unicast routing tables. If the packet was received on the best path back to the source then the router forwards a copy of the packet on all other interfaces. Otherwise, the packet is discarded.

Dense–Mode PIM uses information about the source interface and the outgoing interfaces, to construct a source–based spanning tree. If an interface has no neighbor multicast routers or no group members, then the interface is "pruned" from the tree. For example in Figure 14–2, if Host J left the multicast group, then DECNIS 4 would send a "prune" message to DECNIS 1. DECNIS 1 would then prune its interface to DECNIS 4 and not send multicast packets to DECNIS 4.

Periodically the DECNIS adds any "pruned" interfaces back to its source–based spanning tree and when it next forwards a multicast packet it sends it out on all interfaces. This is because a pruned interface could regain a neighbor multicast router or host members at any time. For example, Host J could rejoin the multicast group.

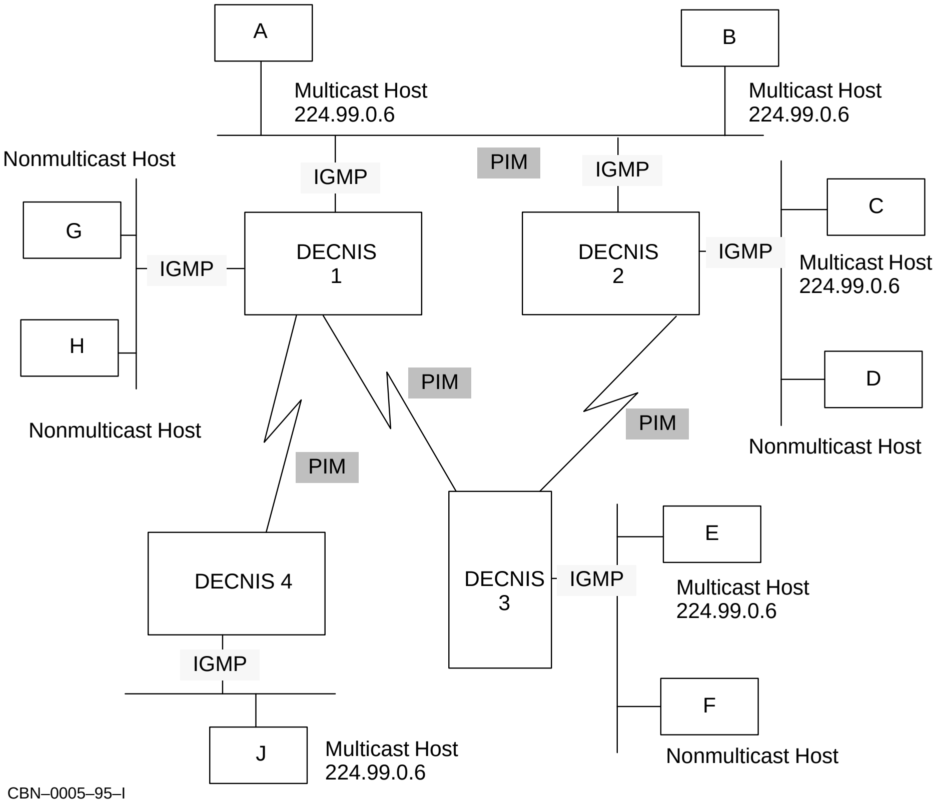
Dense–Mode PIM also uses pruning to avoid duplication of packets. If there are two possible paths to a group member then Dense–Mode PIM prunes anything but the shortest path from its source based tree.

However it is possible for duplicate packets to be delivered initially and periodically, before any pruning has occured. For this reason, multicast applications need to be able to cope with duplicate copies of a packet.

For example in Figure 14–2, if Host E sources a packet, DECNIS 1 and DECNIS 2 will both receive it and forward it onto the LAN to Hosts A and B. This means that initially Hosts A and B receive two copies of the packet.

However when DECNIS 1 and 2 find duplicates on the LAN, they will use Dense–Mode PIM to determine which DECNIS should be used to forward packets onto the LAN. The other DECNIS will then prune its interface to the LAN.

Figure 14–2 IP Multicast Protocols Operating Between DECNIS Systems



14.5 Setting Up IP Multicasting on DECNIS Systems

This section describes how to set up IP Multicasting. Section 14.5.1 describes how to plan your multicast configuration and Section 14.5.2 describes the commands required to set up IP multicasting.

14.5.1 Planning Your IP Multicast Configuration

A DECNIS becomes an IP multicast router when an IGMP entity is created and enabled. Once an IGMP entity is enabled, the PIM protocol is automatically started.

Set Up an IGMP Entity On All the Interfaces of Every DECNIS System

Figure 14–3 shows a correct IP multicasting configuration where IGMP has been set up on all interfaces of all the DECNIS systems.

It is recommended that you create the IGMP entity on all interfaces and all systems because:

- Multicast hosts may join a host group at any time. For example, in Figure 14–3, IGMP is set up on the DECNIS 1 interface to Hosts G and H, as these may become multicast hosts at any time.
- Any router may become part of the shortest path between a multicast source and a host. Dense–Mode PIM always uses the shortest path between a source and a host, regardless of whether the path contains non-PIM routers. If a multicast packet reaches a non-PIM router then the packet is discarded.

Restrictions

The following restrictions apply to IP multicasting on the DECNIS.

- IP multicasting is only supported on DECNIS systems with an MPC-II.
- IGMP is not supported on X.25 DA circuits.
- RFC 1112 does not support IGMP on PPP circuits between routers and hosts.

14.5.2 The Commands Required to Set Up IP Multicasting

14.5.2.1 Creating the IGMP Entity

To create the IGMP entity on an interface, enter the following command:

```
NCL> CREATE ROUTING CIRCUIT circuit-name IGMP
```

where *circuit-name* is the name of the circuit.

For example:

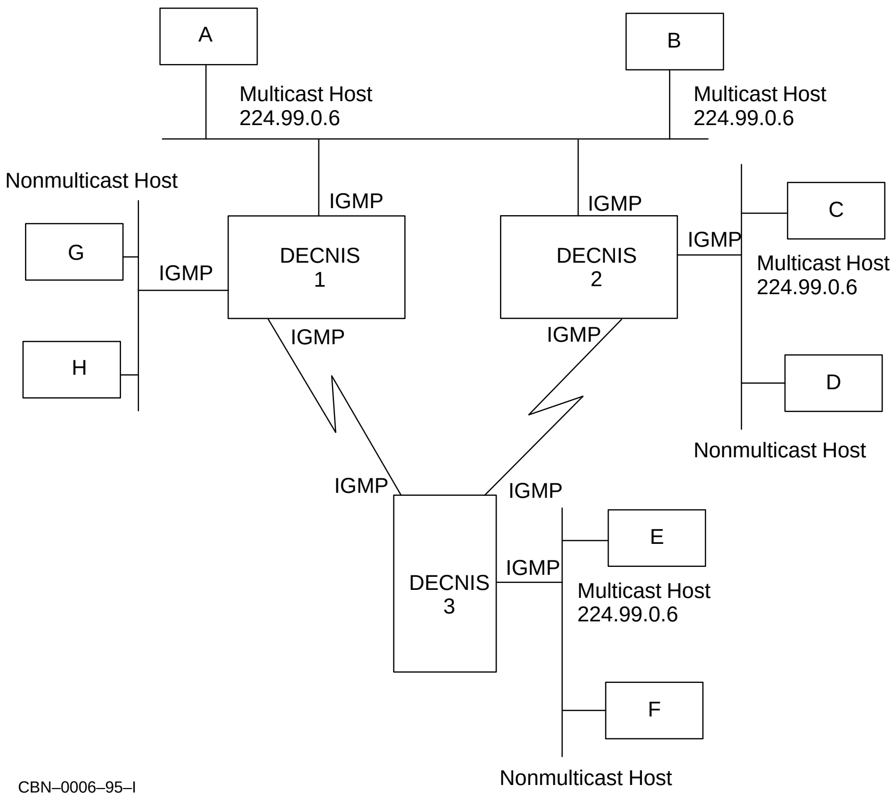
```
NCL> CREATE ROUTING CIRCUIT l602-3-0 IGMP
```

For FDDI circuits only, specify which host groups the DECNIS should listen to by entering the following command:

```
NCL> SET ROUTING CIRCUIT circuit-name IGMP GROUPS {d-address, d-address}
```

where *d-address* is the Class D IP destination address for the multicast host group.

Figure 14–3 An Example IP Multicast Configuration



For example:

```
NCL> SET ROUTING CIRCUIT f621-6-0 IGMP GROUPS {224.99.0.6, 224.99.0.7}
```

14.5.2.2 Enabling the IGMP Entity

Enable the IGMP entity on all interfaces.

Enter the following command:

```
NCL> ENABLE ROUTING CIRCUIT circuit-name IGMP
```

Once IGMP is enabled on a circuit, the PIM protocol is automatically started and the DECNIS starts listening to multicast hosts and routers.

14.5.2.3 Example NCL Commands

In Figure 14–3, DECNIS 3 has 2 WAN interfaces and 1 LAN interface. The following commands are an example of how you might set up DECNIS 3 as a multicast router:

```
NCL> CREATE ROUTING CIRCUIT L601-3-0 IGMP
NCL> CREATE ROUTING CIRCUIT W622-4-0 IGMP
NCL> CREATE ROUTING CIRCUIT F621-6-0 IGMP
NCL> SET ROUTING CIRCUIT F621-6-0 IGMP GROUPS {224.99.0.6}
NCL> ENABLE ROUTING CIRCUIT L601-3-0 IGMP
NCL> ENABLE ROUTING CIRCUIT W622-4-0 IGMP
NCL> ENABLE ROUTING CIRCUIT F621-6-0 IGMP
```

15

IP Standby

This chapter describes the IP Standby facility on the DECNIS.

15.1 Overview

IP Standby is a facility that enables DECNIS IP routers on a LAN to act as backup (or standby) routers to each other. If an IP Standby router fails, or its connection to the LAN fails, another IP Standby router will rapidly take over.

Why IP Standby Is Needed

When an IP router goes down, there can be long delays before IP hosts detect the failure and switch to an alternative router. IP Standby greatly reduces such delays. Normally, the delay between an IP Standby router going down, and the hosts on a LAN switching to the new IP Standby router is about five seconds, but it can be less.

15.2 How IP Standby Works

This section describes how IP Standby works.

15.2.1 "Virtual" MAC Addresses

With IP Standby, you assign a "virtual" MAC (LAN) address to each IP standby router on a LAN. The virtual MAC address is associated with the IP address (or addresses) on the router’s LAN circuit.

To set up IP Standby, you need to:

- Select a "virtual" MAC address for each IP Standby router, from a set of MAC addresses reserved for use with the IP Standby protocol.
- Configure each IP standby router to know the virtual MAC address and associated IP address of all the IP Standby routers on the LAN.

How the Virtual MAC Address Is Used

Once IP Standby is enabled, the router supplies its virtual MAC address (rather than its "real" MAC address) as the source address in ARP responses and ARP requests. This means that it uses the virtual MAC address for receiving IP data.

Note that the virtual MAC address is **not** used as the source MAC address in IP data packets sent by the router to hosts. However, the virtual MAC address is used as the source address in IP Standby Hello messages, so that filtering bridges can learn the correct location of those addresses.

15.2.2 The Primary Router

Once the IP Standby routers are up and running, they elect one of their number as a **Primary router**.

If one of the IP Standby routers fails, the Primary router "impersonates" it by taking over its virtual MAC address, and the associated IP address. If the Primary router fails, a new Primary router is elected, which immediately takes over the virtual MAC address and IP address of the failed router.

Note that all IP Standby routers, whether Primary or not, route host traffic, which they optimize by using ICMP redirect messages.

15.2.3 Example

In Figure 15–1, DECNIS A and B are IP Standby routers on LAN A. You have the following information:

	MAC Address	IP Address	Primary Router?
DECNIS A	4	130.223.14.1	Yes
DECNIS B	3	130.223.14.2	No

In Figure 15–2, DECNIS B has gone down. DECNIS A takes over DECNIS B's virtual MAC address for replying to ARP messages. IP packets previously sent to DECNIS B now go to DECNIS A.

Figure 15–1 Both DECNIS Routers Up

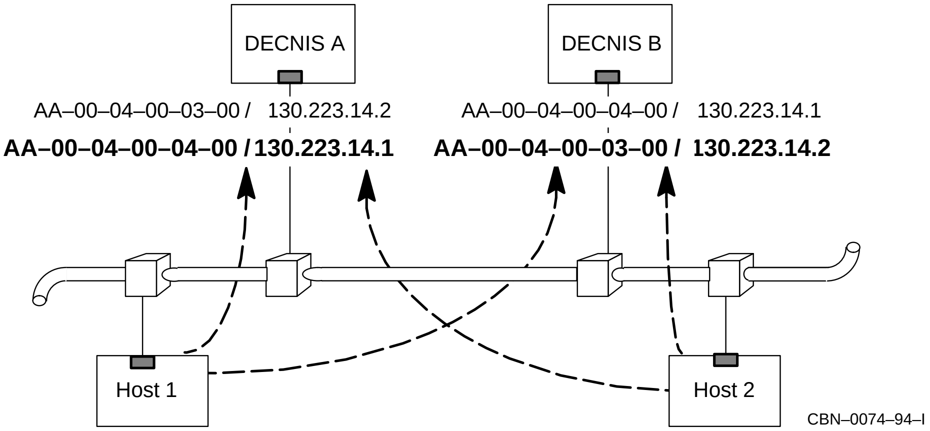
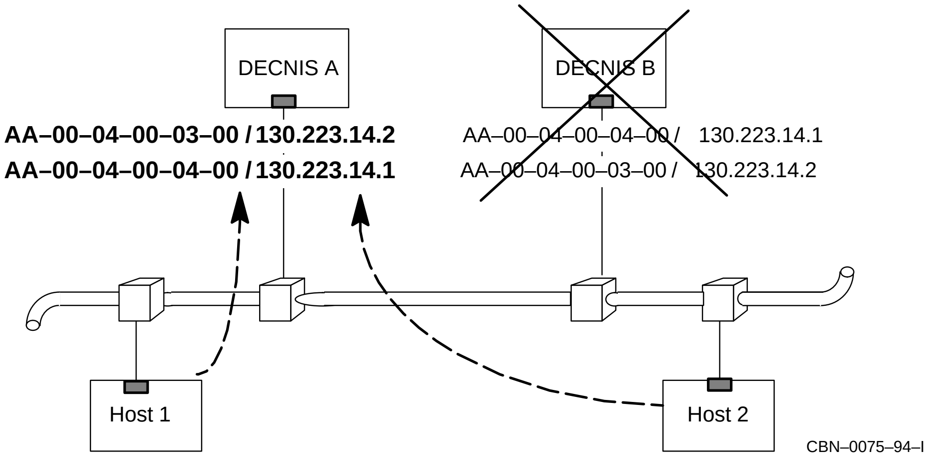

DECNIS A
DECNIS B
AA–00–04–00–03–00 / 130.223.14.2
AA–00–04–00–04–00 / 130.223.14.1
AA–00–04–00–04–00 / 130.223.14.1
AA–00–04–00–03–00 / 130.223.14.2
Host 1
Host 2
CBN–0074–94–I

Figure 15–2 DECNIS B Fails


DECNIS A
DECNIS B
AA–00–04–00–03–00 / 130.223.14.2
AA–00–04–00–04–00 / 130.223.14.1
AA–00–04–00–04–00 / 130.223.14.1
AA–00–04–00–03–00 / 130.223.14.2
Host 1
Host 2
CBN–0075–94–I

15.2.4 Criteria for Remaining an IP Standby Router

Once the IP Standby routers are up and running, they send out one IP Standby Hello message per second. If a router is not doing this, it is considered to be down, and the Primary router takes over its virtual MAC address.

15.2.5 Electing a Primary Router

The IP Standby routers re-elect a Primary router when an IP Standby router fails or recovers, or when a new IP Standby router is added. The criteria for electing a Primary router are:

- Whether all its **monitored circuits** are working. These are circuits that you specify should be monitored for failure.
- Its **priority**. If all the IP Standby routers are the same with respect to monitored circuits—that is, they all have failed circuits, or none has failed circuits—then the Primary router is elected on the basis of priority. You can specify priority when you set up IP Standby.
- If there is more than one DECNIS with the same priority, the Primary router will be the one with the highest virtual MAC address.

Summary of Primary Router Election

Figure 15–3 summarizes the process that the IP Standby routers use to elect a Primary router.

15.2.6 What the Primary Router Does

When an IP Standby router fails, the Primary router "impersonates" it by doing the following:

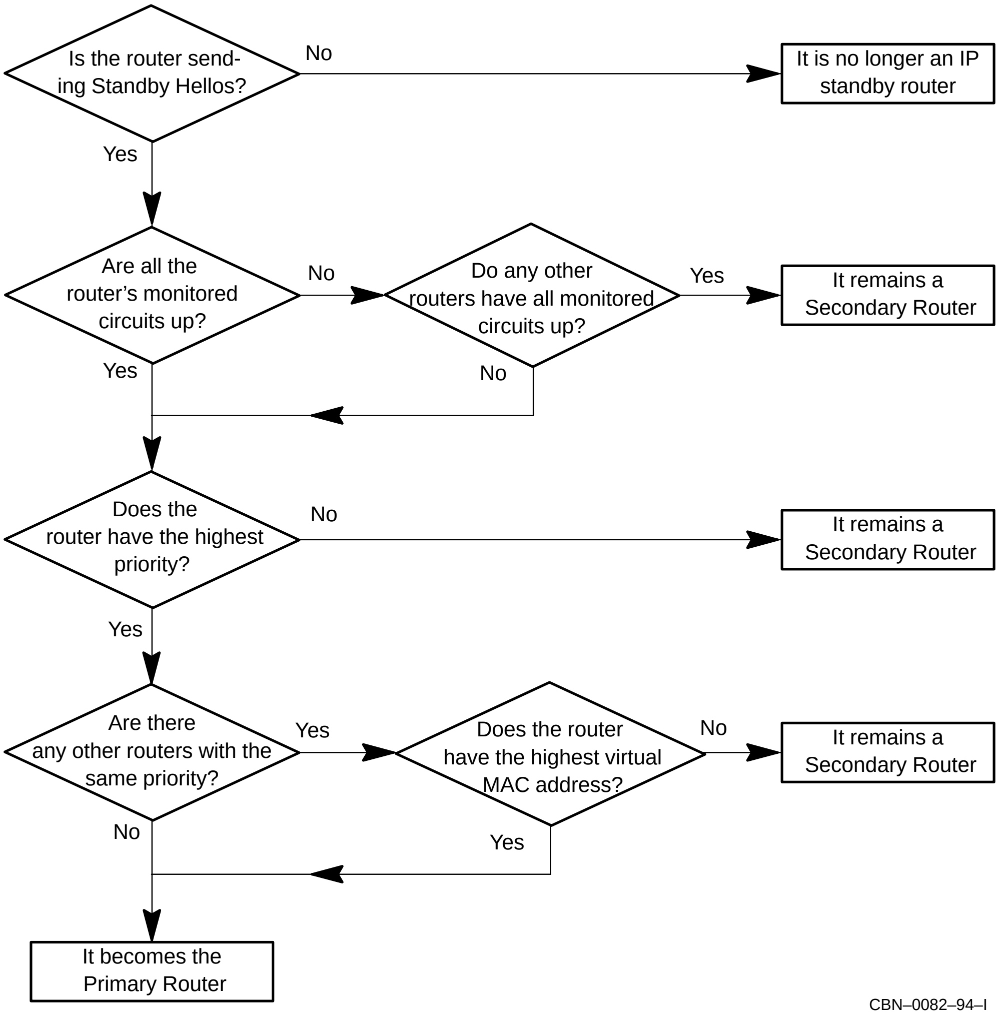
- Responding to ARP requests intended for the failed router’s IP address, giving the failed router’s MAC address as the source address.
- Receiving IP data intended for the virtual MAC address of the failed router.

15.2.7 When a Router Recovers

When a failed IP Standby router recovers and announces itself, the Primary router stops impersonating it.

There is a very brief interval between a router recovering and the Primary router ceasing to use its addresses. In this interval, it is possible that packets may be lost or duplicated.

Figure 15–3 How a Primary Router Is Elected


Is the router sending Standby Hellos?
No
It is no longer an IP standby router
Yes
Are all the router's monitored circuits up?
No
Do any other routers have all monitored circuits up?
Yes
It remains a Secondary Router
Yes
No
Does the router have the highest priority?
No
It remains a Secondary Router
Yes
Are there any other routers with the same priority?
Yes
Does the router have the highest virtual MAC address?
No
It remains a Secondary Router
No
Yes
It becomes the Primary Router
CBN–0082–94–I

15.3 Setting Up IP Standby

This section describes how to set up IP Standby.

15.3.1 Tasks

Table 15–1 lists the tasks involved in setting up IP Standby:

Table 15–1 IP Standby Tasks

On each IP Standby router on the LAN, do this...	For details, refer to...
Create an IP Standby entity on the LAN Routing Circuit	Section 15.3.3
Specify the virtual MAC address/IP address pair of each IP standby router on the LAN	Section 15.3.3
Set up monitored circuits	Section 15.3.4
Specify the IP Standby priority	Section 15.3.5
Specify the time to wait between the failure of a Primary router and the election of a new one	Section 15.3.6

15.3.2 Before You Begin

This section contains information that you need to know about before you set up IP Standby:

15.3.2.1 Type of Routing Circuit Required

You can only configure IP Standby on CSMA/CD or FDDI routing circuits.

15.3.2.2 MAC Addresses and MAC Selectors

A set of sixteen MAC addresses is reserved for use with the IP Standby protocol as virtual MAC addresses. These addresses are the same except for one digit.

Each IP Standby router on an extended LAN must have a different virtual MAC address.

Virtual MAC Address Format

The virtual MAC address is always as follows:

```
AA-00-04-00-0x-00
```

where *x* is any hexadecimal digit.

The MAC Selector

The digit represented by *x* is known as the **MAC selector**. In commands to configure IP Standby, you represent the MAC address by the MAC selector.

15.3.2.3 Problems with Disabling IP Standby

Disabling or enabling IP Standby on an operational router can cause serious problems both for hosts and for remaining IP standby routers on a LAN.

In particular, as long as a router is operational, hosts are unlikely to update their ARP caches to add or delete the router's "virtual" MAC address. This is because a host normally will not update its ARP cache as long as a router continues to accept packets.

Thus, if you enable IP Standby on an operational router, hosts may not update their ARP caches to add the "virtual" MAC address. Similarly, if you disable IP Standby, hosts may continue to use the router's "virtual" MAC address, even though it is no longer valid.

The consequence in both cases is that if the IP Standby router fails, the hosts may not switch to the Primary router for a long time, or at all.

Restriction on Entering Commands

Because of these problems, you should not enter dynamically any of the NCL commands listed below. Instead, enter them in the appropriate user NCL script files (see Section 1.5.6).

- Any NCL command that requires you to disable the IP Standby entity first. These are the commands listed in Section 15.3.3.
- The DISABLE ROUTING CIRCUIT IP STANDBY command.

15.3.2.4 Requirements for IP Addresses

IP addresses for an IP Standby router must meet the following requirements:

- Each IP Standby IP address on a routing circuit must already have been defined as an IP address on the circuit. (Refer to Section 10.5.7 and Section 10.24.3 for information about defining IP addresses on routing circuits.)
- The same subnets must be defined on all IP Standby routers on a LAN. For example, if you set up IP addresses in subnets 4.3.15.0, 4.3.18.0 and 4.3.20.0 on one IP Standby router, you must also set up IP addresses in those subnets on the other IP Standby routers on the LAN.

15.3.3 Procedure

This section gives the NCL commands required to set up IP Standby on a routing circuit. Do not enter these commands dynamically; instead add them to the appropriate user NCL script files (see Section 1.5.6).

```
CREATE ROUTING CIRCUIT circuit-name IP STANDBY MODE MAC ❶
SET ROUTING CIRCUIT circuit-name IP STANDBY ROUTERS -  ❷
{[MAC SEL=n, ADDR={a.b.c.d, e.f.g.d}], -
[MAC SEL=m, ADDR={a.b.c.h, e.f.g.h}]}
ENABLE ROUTING CIRCUIT circuit-name IP STANDBY ❸
```

❶ Create the IP STANDBY entity, and specify MAC mode, where *circuit-name* is the name of the circuit on which you are setting up IP Standby.

Note that the IP STANDBY entity does not have a name. You can have only one IP STANDBY entity per routing circuit.

❷ Specify a MAC selector and list of associated IP addresses for each router in the IP standby set, including the router which is being configured. Note that the list of MAC selector/IP addresses must be exactly the same for each IP Standby router in the IP Standby set.

- MAC SEL is the MAC selector of a router on a LAN. The variables *n* and *m* are MAC selectors.

  Each router must have a unique MAC selector on the LAN.

- ADDR is a list of the router's IP addresses on its interface on the LAN. If the router only has one IP address, then you only need specify one IP address. *a.b.c.d*, *e.f.g.d*, *a.b.c.h*, *e.f.g.h* are IP addresses.

❸ Enable the IP Standby entity on the routing circuit.

15.3.4 Setting Up Monitored Circuits

You can specify particular routing circuits as **monitored circuits**. Such circuits are monitored for failure once per second.

15.3.4.1 Purpose of Monitored Circuits

IP Standby automatically checks the status of the Primary router's IP Standby LAN circuits, to make sure it is still eligible to be a Primary router. The Monitored Circuits feature makes a similar check on other designated routing circuits. This allows the router's eligibility as a Primary router to be influenced by whether the monitored circuits are working. The Monitored Circuits feature can prevent a router from continuing as a Primary router when it cannot reach parts of the network.

15.3.4.2 Result of Configuring Monitored Circuits

If you configure any circuits as monitored circuits, the following happens:

- If all the data links for the monitored circuits are running, then the DECNIS remains a candidate for election as the Primary router.
- If any of the monitored circuit data links fail, then the DECNIS stops being eligible for election as the Primary router. However, there are two exceptions to this:
  - If a monitored circuit is a Primary circuit in a Supervisor Group. In this case, the data link state is ignored, since it is assumed the Secondary circuit will take over in the case of failure. Refer to Section 9.30 for more about Supervisor Groups.
  - If all the routers in a group of IP Standby routers have one or more failed monitored circuits.

15.3.4.3 Cannot Monitor Disabled or Nonexistent Circuits

A circuit is not considered to be failed if it has been disabled. Disabling a monitored circuit will therefore have no effect on whether an IP Standby router is eligible to be a Primary router.

The circuit you specify as a monitored circuit must be a previously created routing circuit. If you enter the name of a nonexistent circuit as a monitored circuit, it will never be considered to have failed.

15.3.4.4 Procedure

To specify circuits to be monitored, enter the following command:

```
NCL> SET ROUTING CIRCUIT circuit-name IP STANDBY -
_NCL> MONITORED CIRCUITS {mon-circ-1,mon-circ-2} -
```

where *circuit-name* is the name of the IP Standby LAN circuit, and *mon-circ-1* and *mon-circ-2* are the names of the circuits being monitored for failure.

15.3.5 Setting Priority

You can determine whether a particular router is likely to be selected as Primary by giving the PRIORITY attribute a high value.

15.3.5.1 Procedure

To set the priority for an IP Standby circuit, enter the following command:

```
NCL> SET ROUTING CIRCUIT circuit-name IP STANDBY PRIORITY n
```

where *n* is a decimal number in the range 1 to 127. The default is 64. The higher the number, the higher the priority.

15.3.6 Setting the Holding Multiplier

By default, an IP Standby router assumes another IP Standby router has gone down if no IP Standby Hello message is received from it after 3 seconds. You can reduce or extend this period by setting the IP STANDBY HOLDING MULTIPLIER.

This is a multiplier of the period between IP Standby Hello messages, which is fixed at one second. By default, the value of the HOLDING MULTIPLIER is 3. The minimum value is 2.

The default value is set low, as a low HOLDING MULTIPLIER value enables IP Standby router failure to be detected more quickly. However, on an exceptionally busy network, or a very large extended LAN, you may want to increase this value, for example, to 5.

15.3.6.1 Procedure

To set the holding multiplier, enter the following command:

```
NCL> SET ROUTING CIRCUIT circuit-name IP STANDBY HOLDING MULTIPLIER n
```

where *n* is a decimal number in the range 2 to 63 inclusive.

15.3.7 IP Standby Example

Figure 15–4 shows an example LAN, with its IP Standby routers.

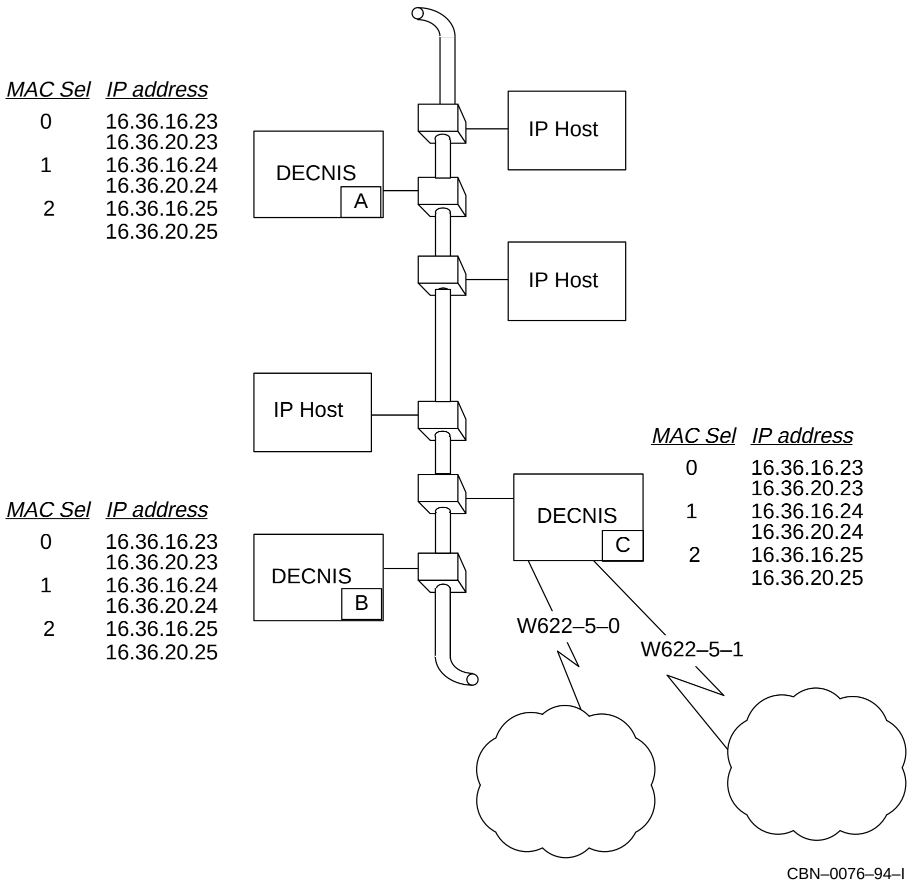
Table 15–2 Description of the Example LAN

	DECNIS A	DECNIS B	DECNIS C
IP Standby LAN Circuits	L601-3-0	L601-3-0	L602-4-0
MAC selectors	0	1	2
IP addresses	16.36.16.23 16.36.20.23	16.36.16.24 16.36.20.24	16.36.16.25 16.36.20.25
Monitored circuits			W622-5-0 W622-5-1

You want to achieve the following:

- Set up all the DECNIS routers as IP Standby routers for each other.
- On DECNIS C, monitor the WAN circuits W622-5-0 and W622-5-1.

Figure 15–4 IP Standby Example



15.3.7.1 Procedure

Enter the NCL commands given below for each DECNIS in the appropriate user NCL script files.

DECNIS A

```
CREATE ROUTING CIRCUIT L601-3-0 IP STANDBY MODE MAC
SET ROUTING CIRCUIT L601-3-0 IP STANDBY ROUTERS -
{[MAC SEL=0, ADDR={16.36.16.23, 16.36.20.23}], -
[MAC SEL=1, ADDR={16.36.16.24, 16.36.20.24}], -
[MAC SEL=2, ADDR={16.36.16.25, 16.36.20.25}]}
ENABLE ROUTING CIRCUIT L601-3-0 IP STANDBY
```

DECNIS B

```
CREATE ROUTING CIRCUIT L601-3-0 IP STANDBY MODE MAC
SET ROUTING CIRCUIT L601-3-0 IP STANDBY ROUTERS -
{[MAC SEL=0, ADDR={16.36.16.23, 16.36.20.23}], -
[MAC SEL=1, ADDR={16.36.16.24, 16.36.20.24}], -
[MAC SEL=2, ADDR={16.36.16.25, 16.36.20.25}]}
ENABLE ROUTING CIRCUIT L601-3-0 IP STANDBY
```

DECNIS C

```
CREATE ROUTING CIRCUIT L602-4-0 IP STANDBY MODE MAC
SET ROUTING CIRCUIT L601-3-0 IP STANDBY ROUTERS -
{[MAC SEL=0, ADDR={16.36.16.23, 16.36.20.23}], -
[MAC SEL=1, ADDR={16.36.16.24, 16.36.20.24}], -
[MAC SEL=2, ADDR={16.36.16.25, 16.36.20.25}]}
SET ROUTING CIRCUIT L602-4-0 IP STANDBY -
MONITORED CIRCUITS {W622-5-0, W622-5-1}
ENABLE ROUTING CIRCUIT L602-4-0 IP STANDBY
```

15.4 Setting Routing Timers for IP Standby

To get the maximum advantage from IP Standby, you should adjust the values of certain routing timers. This section describes how to do this.

15.4.1 Introduction

When an IP Standby router fails, or recovers, it is important for other IP Standby routers to notice this as quickly as possible, so that the Primary router can take over.

The time required for IP Standby routers to decide that another router has failed is partly governed by the IP Standby timers described in Section 15.3.6. However, it is also affected by various routing timers. If these routing timers are set to a low value, routers will quickly learn and propagate information about failed routers.

15.4.1.1 Why You Need to Adjust Routing Timers

The IP Standby mechanism on a router works independently of the routing tables. Although a Primary router takes over very quickly when another router fails, if routing timer values are high the other routers will not immediately detect that a failed router is no longer the best path to a destination.

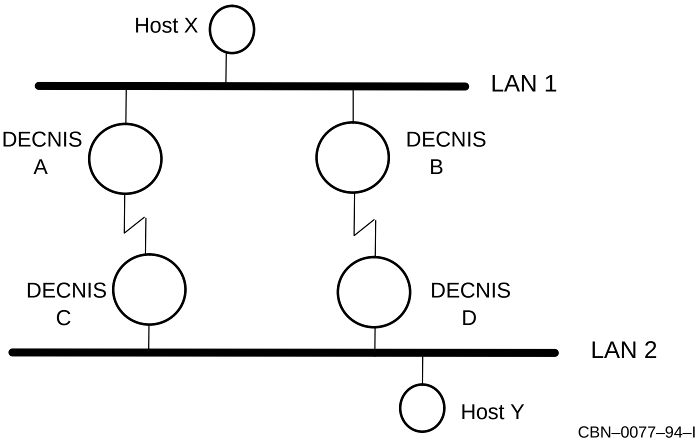
Example

In Figure 15–5, DECNIS A, B, C and D are IP Standby routers, with point-to-point links connecting LAN 1 and LAN 2. The link between DECNIS A and DECNIS C is set up as the normal route between the LANs.

If DECNIS A goes down, DECNIS B becomes the Primary router; traffic from host X to host Y is now handled by DECNIS B. However, because DECNIS C and D's routing timers are set to high values, reverse direction traffic is lost; this is because DECNIS C will continue to attempt to send traffic to DECNIS A until the timers expire.

Similarly, if the link from DECNIS B to DECNIS D was set to an unusually high cost, DECNIS B might continue to attempt to send traffic to DECNIS A, until it detected that this was no longer a viable route.

Figure 15–5 Example



15.4.1.2 Routing Timer Tradeoffs

Although reducing timer values improves the effectiveness of IP Standby, it also has other effects, which you should take into account:

- It increases the amount of control traffic on the network.
- It makes the network more sensitive to transient failures and congestion.

15.4.1.3 Routing Timers

To get optimum IP Standby performance, adjust the following routing timers:

Routing Timer	Refer to...
Routing Hello timers	Section 15.4.2
Routing holding multipliers	Section 15.4.2
Minimum interval for link state packet generation	Section 15.4.3

15.4.2 Setting Hello Timers and Holding Multipliers

The Hello timer on a routing circuit sets the number of seconds between Hello messages on that circuit.

Routing holding multipliers set the number of Hello messages that can be lost before a router is regarded as failed.

Table 15–3 shows the Routing hello timer and holding multiplier attributes you can adjust to optimize IP Standby performance. If you want the IP Standby routers to detect quickly that another router has gone down, set Hello timers and holding multipliers on all connected routers to the values recommended in the table.

Note that the attributes are different for LAN circuits than for point-to-point circuits.

Table 15–3 Routing Timers

Parameter	Default	Recommended
LAN circuits:		
ROUTING CIRCUIT ISIS HELLO TIMER	3	1
ROUTING ISIS HOLDING MULTIPLIER	10	2
Point-to-Point circuits:		
ROUTING CIRCUIT HELLO TIMER	10	1
ROUTING HOLDING MULTIPLIER	3	2

15.4.2.1 Procedure: LAN Circuits

This section shows how to set the timers for the LAN circuits on the IP Standby router.

Routing Holding Multiplier

To set the routing holding multiplier to the recommended value, you need to enter the following command in the user NCL SET script file. Do not enter this command dynamically.

```
SET ROUTING ISIS HOLDING MULTIPLIER 2
```

Note that this sets the holding multiplier for all LAN circuits.

Hello Timer

To set the routing circuit hello timer on the LAN circuits to the recommended value, enter the following command for each LAN circuit:

```
NCL> SET ROUTING CIRCUIT circ-name ISIS HELLO TIMER 1
```

where *circ-name* is the name of the routing circuit.

15.4.2.2 Procedure: Point-to-Point Circuits

This section explains how to set the routing timers on the point-to-point circuits on the IP Standby router.

Holding Multiplier

To set the routing holding multiplier to the recommended value, enter the following command in the user NCL SET script file. Do not enter this command dynamically.

```
SET ROUTING HOLDING MULTIPLIER 2
```

This sets the holding multiplier for all point-to-point circuits.

Hello Timer

To set the routing circuit hello timer on point-to-point circuits to the recommended value, enter the following command:

```
NCL> SET ROUTING CIRCUIT circ-name HELLO TIMER 1
```

where *circ-name* is the name of the routing circuit.

15.4.3 Rerunning the Routing Decision Process

The speed with which routers adapt to the failure of other routers is partly governed by the ROUTING attribute MINIMUM LSP GENERATION INTERVAL. This is the number of seconds a router must wait before generating a new link state packet.

Routers generate new link state packets in response to network events, such as a node going down. When a router sends a link state packet, it causes other routers to rerun their routing decision process. The ROUTING MINIMUM LSP GENERATION INTERVAL attribute in effect limits the frequency with which routers on the network run their decision process.

15.4.3.1 Procedure

It is recommended that you set the MINIMUM LSP GENERATION INTERVAL on IP Standby routers, and connected routers, to 1.

To do this, enter the following command in the user NCL SET script file. Do not enter this command dynamically.

```
SET ROUTING MINIMUM LSP GENERATION INTERVAL 1
```

15.5 Low-Cost Paths

A low-cost path is an additional LAN link between two IP Standby routers, which is set to have a lower cost than the main LAN connecting the routers to the hosts. For example, it could be an additional Ethernet in parallel with an existing one.

When a WAN link fails, the routers will use the low-cost path for host traffic, rather than sending an ICMP redirect message to the host to cause it to send traffic directly to the other router. (Note that in cases of router failure, IP Standby ensures that the host can continue to use the failed router's address in packets to be forwarded.)

This technique might be useful in the following cases:

- To avoid redirect messages. You might want to do this because:
  - You wish to avoid the redirect message load on the network.
  - Hosts do not implement the processing of ICMP redirect messages. However, note that such hosts do not conform to RFC 1122.
- If hosts have been set up to share traffic between the routers, and the network cost metrics would not be sufficiently different to re-establish the required traffic splitting after a WAN link failure.

16

DECnet/OSI Routing

16.1 Introduction

This chapter describes how to manage DECnet/OSI routing on the DECNIS. It includes information on running the DECnet Phase IV routing protocols.

16.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

Note

For any tasks that require you to disable a routing circuit, ensure that you do not disable the routing circuit that you are using to manage the DECNIS, unless there is an alternative route between the host on which you issue the NCL commands and the DECNIS.

16.2 DECNIS Addressing

16.2.1 Introduction

Network addresses in a DECnet/OSI network are called NSAP (Network Service Access Point) addresses.

The DECNIS can have from one to three NSAP addresses.

16.2.2 DECNIS Addresses and DECnet Phase IV Nodes

If the DECNIS is in a network that has DECnet Phase IV nodes, one of its NSAP addresses must be in a Phase IV-compatible form.

16.2.3 Address Structure

The structure of a DECNIS NSAP address is as follows:

Area_Address:*System_ID*:20

where the values of *Area_Address* and *System_ID* depend on whether the NSAP address is to be Phase IV-compatible.

16.3 Deriving the Phase IV-Compatible NSAP Address

16.3.1 Introduction

If the DECNIS is in a network with any Phase IV nodes, one of its NSAP addresses must be in a Phase IV-compatible form.

The DECNIS constructs its Phase IV-compatible NSAP address from information you supply during configuration.

16.3.2 Procedure

Derive the Phase IV-compatible NSAP address as follows:

1. Convert the area part of the Phase IV address into hexadecimal.

   Example: If the Phase IV address is 13.7, convert the 13 to hexadecimal, giving 0D.

2. Derive the area address by combining the Phase IV prefix with the previous result as follows:

   Area_Address = *PhaseIV_prefix*00-*PhaseIV_area*

   Example: If the Phase IV prefix is 49::, the area address will be:

   ```
   49::00-0D
   ```

3. Given that the Phase IV address is $n.m$, enter the values for n and m in the following equation:

   $$(n * 1024) + m = total$$

   Then convert the value *total* into hexadecimal.

   Example:

   $$(13 * 1024) + 7 = 13319 = 00003407 \text{ (hex)}$$

4. Derive the system-ID using the following format:

   ```
   AA-00-04-00-xx-yy
   ```

   where *xx* represents the last two digits of the hexadecimal value *total* and *yy* represents the two digits previous to the last two digits of *total*.

   Example:

   ```
   AA-00-04-00-07-34
   ```

   Note that in the system-ID, the two pairs of digits from the value *total* are reversed in order.

5. Derive the NSAP address by combining the area address and the system-ID as follows:

   ```
   PhaseIV_Area_Address:System_ID:20
   ```

   Example:

   ```
   49::00-0D:AA-00-04-00-07-34:20
   ```

   where 49::00-0D is the area address for the DECNIS and AA-00-04-00-07-34 is the system-ID for the DECNIS.

16.4 Changing the Phase IV Prefix

16.4.1 Restrictions

Reboot the DECNIS

You cannot change the Phase IV prefix of a running system. Instead, you must change the Phase IV prefix within the DECNIS configurator (Window or text-based), as described in Section 16.4.2 and Section 16.4.3, and then reboot the DECNIS.

Use the Same Phase IV Prefix Throughout the Network

You must give all the DECNIS systems in your network the same Phase IV prefix.

16.4.2 Procedure: DECNIS Text-Based Configurator

To change the Phase IV prefix of the DECNIS within the DECNIS text-based configurator, follow the steps show below, and then reboot the DECNIS.

	Screen	Action
1	Sections Menu	Select Routing.
2	Routing Options Menu	Select Modify Routing Information.
3	Routing	Ensure that you have answered Yes to the question "Do you wish to supply a Phase IV address?" Note that you will only see this question if you have chosen not to run the Phase IV routing algorithm at either Level 1 or Level 2.
4		Enter the new Phase IV address prefix.

16.4.3 Procedure: clearVISN DECNIS Configurator

To change the Phase IV prefix of the DECNIS within the clearVISN DECNIS configurator follow these steps:

1. On the Main Navigation window, click **DNA IV/V + OSI**.
2. On the Addressing tab page, delete the current Phase IV prefix and enter the new one.
3. Click **OK**.
4. On the Main Navigation window, click **Compile**.

16.5 Changing the Phase IV Address

16.5.1 Reboot the DECNIS

You cannot change the Phase IV address of a running system. You must change the Phase IV address within the configurators, as described in Section 16.5.2 and Section 16.5.3, and reboot the system.

16.5.2 Procedure: DECNIS Text-Based Configurator

To change the Phase IV address of the DECNIS within the DECNIS text-based configurator, follow these steps:

1. Run the load-host configurator to enter the new Phase IV address.
2. Run the DECNIS text-based configurator to generate a new master NCL script file containing the new Phase IV address.
3. Reboot the DECNIS.

16.5.3 Procedure: clearVISN DECNIS Configurator

To change the Phase IV address of the DECNIS within the clearVISN DECNIS configurator follow the steps shown below.

1. On the Main Navigation window, click **DNA IV/V + OSI**.
2. On the Addressing tab page, delete the current Phase IV address and enter the new one.
3. Click **OK**.
4. On the Main Navigation window, click **Compile**.

   This compiles the NCL script into a CMIP file and (if specified) creates a combined file.

16.6 Deriving the Phase V-Style NSAP Address

16.6.1 Introduction

The DECNIS constructs its Phase V-style NSAP addresses from information you supply during configuration.

16.6.2 Procedure

Derive the Phase V-style NSAP address as follows:

```
Area_Address:System_ID:20
```

where *Area_Address* is a defined area address for the DECNIS and *System_ID* is the base hardware address for the DECNIS.

16.7 Changing the Phase V Area Address

16.7.1 Introduction

To change a Phase V area address of the DECNIS, remove the old address and add the new one.

You can do this without rebooting the DECNIS.

16.7.2 Procedure

Add and remove Phase V area addresses to the DECNIS as follows:

1. Add addresses to the MANUAL AREA ADDRESSES characteristic of the ROUTING entity by using the ADD command:

   ```
   NCL> ADD ROUTING MANUAL AREA ADDRESSES {area-address}
   ```

2. Delete addresses from the MANUAL AREA ADDRESSES characteristic of the ROUTING entity by using the REMOVE command:

   ```
   NCL> REMOVE ROUTING MANUAL AREA ADDRESSES {area-address}
   ```

16.8 NSAP Formats

16.8.1 Introduction

The above sections describe the Digital format for NSAP addresses.

NSAP addresses may also be expressed in OSI or HRPF (Hexadecimal Reference Published Format).

16.8.2 OSI Format

In this format, the IDP field of the NSAP is separated from the DSP field by a plus sign, but the fields within the DSP are not separated. Padding digits must therefore be included. ISO 8348/AD2 describes the ISO format for NSAP addresses.

16.8.3 HRPF Format

In this format, there are no symbols to separate the fields of the NSAP, and so all padding digits must be included. HRPF format NSAPs begin with a / character.

16.9 Deciding Which Routing Algorithms to Run

16.9.1 Introduction

The DECNIS can run one of two routing algorithms:

- DECnet/OSI link state
- DECnet Phase IV routing vector

If the DECNIS is configured as a Level 2 router, it can run one routing algorithm at Level 1, and the other at Level 2.

16.9.2 Procedure

Table 16–1 describes briefly how to choose the routing algorithm that the DECNIS should use.

Table 16–1 How to Choose the Routing Algorithms

If...	Then...
There are DECnet Phase IV routers in the same area as the DECNIS...	The DECNIS must run the routing vector algorithm at Level 1.
All the routers in the same area as the DECNIS are running link state routing...	The DECNIS must run link state routing at Level 1.
The DECNIS communicates only with Level 2 routers running the routing vector algorithm...	The DECNIS should run the routing vector algorithm at Level 2.
The DECNIS communicates only with Level 2 routers running the link state routing algorithm...	The DECNIS should run link state routing at Level 2.
The DECNIS communicates at Level 2 with routers running the link state routing algorithm and with routers running the routing vector algorithm...	The DECNIS should run link state routing at Level 2, and use manual routing to communicate with those Level 2 routers running the routing vector algorithm (see Section 16.17).

16.9.3 Phase IV Cluster Aliases

A DECNIS running link state routing can coexist on a LAN with routing vector routers that are part of a Phase IV OpenVMS cluster.

The DECNIS will send out routing vectors indicating one-hop connectivity for all nodes, so as to distract traffic from the Phase IV cluster routers. These routes are advertised with a cost of 714 and a hop count of 30 to prevent them being propagated off the LAN.

16.10 Changing the Routing Algorithms

16.10.1 Consequences of Changing the Routing Algorithms

Changing the routing algorithm at either Level 1 or Level 2 will have serious consequences for your configuration. In particular, the DECNIS text-based configurator will delete any IP route propagation you have entered.

If, as a result of your change, the DECNIS is not running the link state routing algorithm at Level 1 or Level 2, any Phase V addresses will be deleted.

Other changes will depend on your particular configuration.

16.10.2 Reboot the DECNIS

You cannot change the routing algorithm of a running system. Instead, you must change the routing algorithm within the DECNIS configurator (Window or text-based), as described in Section 16.10.3 and Section 16.10.4, and then reboot the DECNIS.

16.10.3 Procedure: DECNIS Text-Based Configurator

To change the DECNIS's routing algorithm(s) within the DECNIS text-based configurator, follow the steps shown below, and then reboot the DECNIS.

	Screen	Action
1	Sections Menu	Select Routing.
2	Routing Options Menu	Select Modify Routing Information.
3	Routing	Select the routing algorithm(s) you wish to run.

16.10.4 Procedure: clearVISN DECNIS Configurator

To change the DECNIS's routing algorithm within the clearVISN DECNIS configurator follow these steps:

1. On the Main Navigation window, click **DNA IV/V + OSI**.
2. On the General tab page, click the routing algorithm you want.
3. Click **OK**.
4. On the Main Navigation window, click **Compile**.

   This compiles the NCL script into a CMIP file and (if specified) creates a combined file.

16.11 Changing the Circuit Cost

16.11.1 Introduction

Each routing circuit has a Level 1 cost and, if applicable, a Level 2 cost associated with it (Section 9.24 describes routing levels). These costs are used by the routing algorithms in deciding which route through the network a packet should take.

16.11.2 Restrictions on Circuit Costs

Different Costs at Each End of a Circuit

It is possible for a circuit connecting two systems to have a different cost at each end: this will mean that traffic will be more likely to travel in one direction over the circuit than the other. This could cause difficulties when diagnosing problems on the network.

Maximum Circuit Cost

Note that the cost of a route, from source to destination, cannot exceed 1023.

16.11.3 Procedure

Set the Level 1 and Level 2 circuit costs of a routing circuit as follows:

1. Set the Level 1 routing cost:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name L1 COST n
   ```

   where *n* is a decimal number between 1 and 63.

2. Set the Level 2 routing cost:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name L2 COST n
   ```

   where *n* is a decimal number between 1 and 63.

16.12 Circuit Priority and Designated Routers

16.12.1 Designated Router Definition

The designated router for a LAN is the router whose routing circuit priority is highest.

16.12.2 Designated Routers and Routing Level

A circuit on a Level 2 router has both a Level 1 priority and a Level 2 priority.

A Level 2 router can be the designated router at one level but not at the other; for example, it could be the designated router for Level 2 traffic, but Level 1 traffic could use a different router.

16.12.3 Function of Designated Level 1 Router

The designated Level 1 router on a LAN is the Level 1 router that generates link state packets containing information about the LAN end nodes.

16.12.4 Function of Designated Level 2 Router

The designated Level 2 router on a LAN is the Level 2 router that generates link state packets containing information about the Level 1 routers on the LAN, together with any manual routing information.

16.13 Changing the Circuit Priority

16.13.1 Introduction

This section describes how to change the Level 1 and Level 2 priority of a routing circuit.

If your DECNIS is not a Level 2 router: do not set the Level 2 routing priority.

If your DECNIS is a Level 2 router but the routing circuit is set to disallow Level 1 traffic: you do not need to set the Level 1 routing priority.

16.13.2 Procedure

Set the Level 1 and Level 2 routing priority as follows:

1. Set the Level 1 routing priority:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name L1 ROUTER PRIORITY n
   ```

   where *n* is a decimal number between 1 and 127.

2. Set the Level 2 routing priority:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name L2 ROUTER PRIORITY n
   ```

   where *n* is a decimal number between 1 and 127.

16.14 Translating DECnet Phase IV Format Packets

The Routing Circuit attribute TRANSLATEIVTOV determines whether the DECNIS translates DECnet Phase IV packets to DECnet/OSI format when the target is an OSI node on the same LAN. This section describes how to use this attribute.

16.14.1 Overview

If the DECNIS receives a DECnet Phase IV format packet on a LAN circuit, and the packet is to be forwarded to an OSI node on the same LAN circuit, there are two possible actions it can take:

- It can forward the packet in Phase IV format. This improves network performance, as it allows the DECnet Phase IV "On-NI" optimization to take place.

- It can translate the packet to DECnet/OSI format. This may be necessary because some OSI nodes cannot read Phase IV packets.

You use the TRANSLATEIVTOV attribute to choose which of these actions you want the DECNIS to take.

Note that if the DECNIS is forwarding Phase IV packets to a DECnet Phase IV node, it always retains the Phase IV format.

16.14.2 Setting the TRANSLATEIVTOV Attribute

This section describes how to set the TRANSLATEIVTOV attribute.

16.14.2.1 Requirement

You can only set this attribute on CSMA/CD or FDDI circuits.

This is because this attribute only applies to DECnet Phase IV packets forwarded on the same circuit on which they are received, and this type of forwarding is only done on LAN circuits.

16.14.2.2 Possible Values

The possible values for the TRANSLATEIVTOV attribute are as follows:

If you choose this value...	The DECNIS will do this...
IF REQUIRED	Always keep DECnet Phase IV format when forwarding to OSI nodes on the same LAN
IF POSSIBLE	Always translate to DECnet/OSI format when forwarding to OSI nodes on the same LAN

Procedure

To set the TRANSLATEIVTOV attribute, follow these steps:

1. Disable the routing circuit:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit-name
   ```

2. Set the TRANSLATEIVTOV attribute:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name TRANSLATEIVTOV value
   ```

   where *value* is either IF POSSIBLE or IF REQUIRED. The default is IF REQUIRED.

3. Enable the routing circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

Alternatively, you can add the command in step 2 to the user NCL script file, NIS_*client-name*_EXTRA_SET.NCL (see Section 1.5.6 for details).

16.15 Manual Routing

16.15.1 Definition

Manual routing is used on routing circuits over which adaptive routing information is not exchanged. You provide the routing information, and change it manually if the network changes.

16.15.2 When to Use Manual Routing

Manual routing must be used in the following circumstances:

- Circuits connecting to foreign routing domains, including X.25 Dynamically Assigned circuits: see Section 16.16.
- Circuits on routers running the link state routing algorithm connecting at Level 2 to routers running the routing vector algorithm: see Section 16.17.
- CSMA/CD routing circuits connecting to end systems that do not support automatic configuration as defined by ISO 9542: see Section 16.19.

16.16 Connecting to Foreign Routing Domains

16.16.1 What Is a Foreign Routing Domain?

A **routing domain** is a collection of end systems, routers and subnetworks that operate the same routing procedures. A **foreign routing domain** is a routing domain that does not or cannot exchange adaptive routing information with the routing domain containing the DECNIS.

16.16.2 Configuration Restriction

Only Level 2 routers can connect to foreign routing domains.

16.16.3 Procedure

Follow these steps to allow a routing circuit to connect to a system in another domain.

1. Disable the routing circuit:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit-name
   ```

2. Set the DNA NEIGHBOR characteristic to FALSE (unless the routing circuit is of type X25 DA):

   ```
   NCL> SET ROUTING CIRCUIT circuit-name DNA NEIGHBOR FALSE
   ```

   Result: This prevents DNA™ routing information from being sent over the circuit. In this way, even if the foreign domain is using DNA protocols, the two domains are kept separate.

Note: You cannot run IP routing on an HDLC or DDCMP routing circuit if you have set DNA NEIGHBOR to FALSE.

3. Create a reachable address for each foreign domain to be reached by this circuit. Specify an address prefix for each reachable address.

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name -
   _NCL> REACHABLE ADDRESS reachable-address ADDRESS PREFIX prefix
   ```

   where *reachable-address* is a name to identify the reachable address, and *prefix* is sufficient leading digits of the NSAP address(es) of the system(s) in the foreign domain to uniquely identify them.

   NSAP address format: Note that you may enter *prefix* in Digital or OSI format (see Section 16.8).

4. Specify the cost of reaching systems in the foreign domain:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> REACHABLE ADDRESS reachable-address COST n
   ```

   where *n* is a decimal number between 1 and 63.

5. If required, set the metric type of the reachable address to External (the default is Internal):

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> REACHABLE ADDRESS reachable-address METRIC TYPE EXTERNAL
   ```

   Set the metric type to External if you want to prevent the reachable address being used to establish a path between two systems in your local routing domain.

6. If the system to which the routing circuit connects is a DECnet Phase IV router, set the data format of the reachable address to Phase IV:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> REACHABLE ADDRESS reachable-address DATA FORMAT PHASEIV
   ```

   Leave the data format at its default (Phase V) if the routing circuit connects to a DECnet/OSI router.

7. If the circuit type is X25 DA, specify the remote DTE address(es) to be used to connect with an address specified by the address prefix of the reachable address:

   - Specify that the remote DTE addresses are to be entered manually:

     ```
     NCL> SET ROUTING CIRCUIT circuit-name -
     _NCL> REACHABLE ADDRESS reachable-address MAPPING MANUAL
     ```

- Now enter the remote DTE addresses:

  ```
  NCL> SET ROUTING CIRCUIT circuit-name
  _NCL>  REACHABLE ADDRESS reachable-address -
  _NCL> DTE ADDRESSES {DTE-addresses}
  ```

8. If the circuit type is CSMA-CD, specify the LAN address to be used to connect with an address specified by the address prefix of the reachable address:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name
   _NCL> REACHABLE ADDRESS reachable-address LAN ADDRESS XX-XX-XX-XX-XX-XX
   ```

9. Enable the reachable address:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name -
   _NCL> REACHABLE ADDRESS reachable-address
   ```

10. Enable the circuit:

    ```
    NCL> ENABLE ROUTING CIRCUIT circuit-name
    ```

16.17 Connecting to Routers Running a Different Routing Algorithm

16.17.1 Introduction

This section describes how to configure a DECNIS that is running the link state routing algorithm at Level 2 to communicate with routers in another area that are running the routing vector routing algorithm at Level 2.

16.17.2 Definition: Interphase Link

An interphase link is a a specific type of manual routing.

An interphase link exists on a DECNIS routing circuit when the circuit has reachable addresses associated with it as follows:

- An outbound reachable address for each area in the routing vector domain that is to be reached through the circuit.
- An inbound reachable address for each area in the link state domain that is to be reached through the circuit.

16.17.3 Outbound Reachable Addresses

The DECNIS uses outbound reachable addresses to inform systems in the link state domain about the systems it can reach in the routing vector domain using the specified circuit.

16.17.4 Inbound Reachable Addresses

The DECNIS uses inbound reachable addresses to inform systems in the routing vector domain about the systems it can reach in the link state domain using its other routing circuits.

16.17.5 Restriction: Phase IV Area Numbers

The two areas connected by an interphase link must have different Phase IV area numbers.

16.17.6 Example

In Figure 16–1, the Level 2 routers in Area 1 and Area 2 communicate using the link state algorithm, and the Level 2 routers in Area 9 and Area 10 communicate using the routing vector algorithm.

In order for the link state domain (Area 1 and Area 2) and the routing vector domain (Area 9 and Area 10) to communicate, you need to set up an interphase link between the DECNIS and the DECnet Phase IV router.

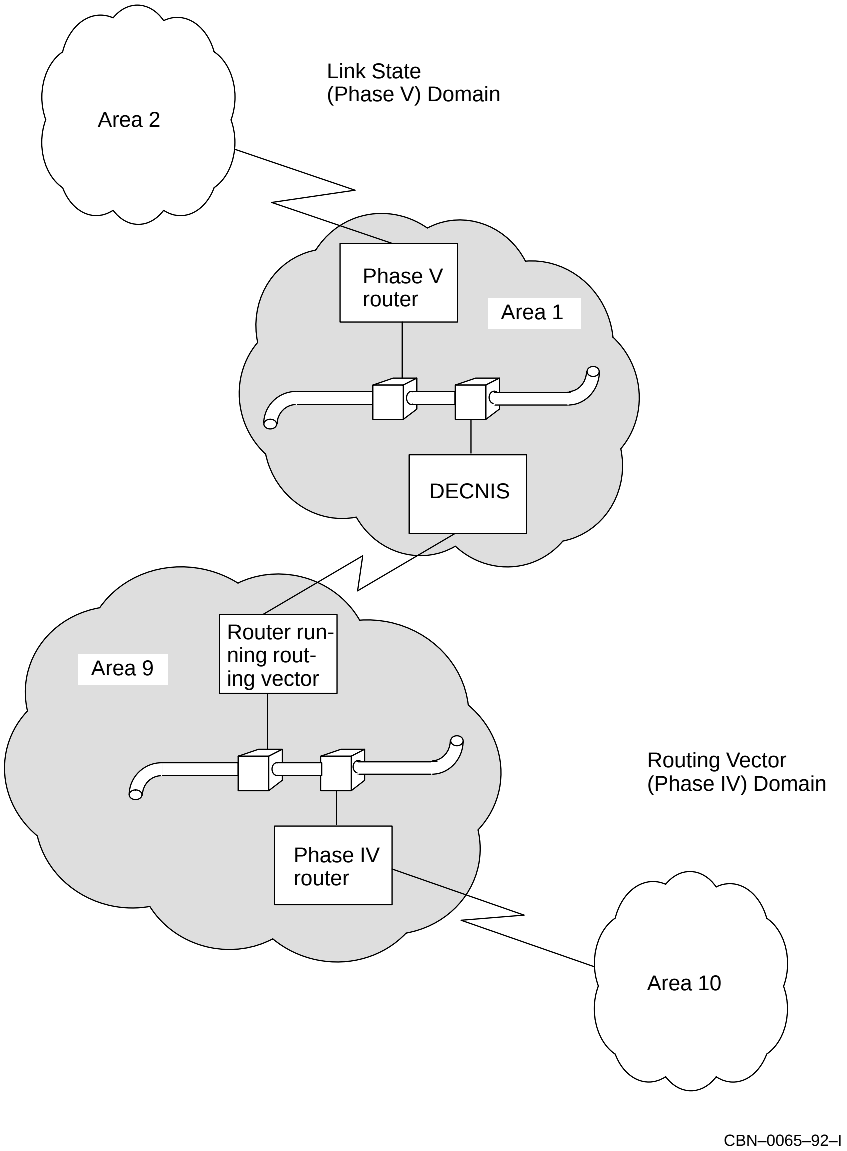
16.17.7 Setting Up an Interphase Link

Follow these steps to set up an interphase link:

1. Create an outbound reachable address for each area in the routing vector domain.

   - Specify the ADDRESS PREFIX for the reachable address; this is the Phase IV compatible area address as described in Section 16.3. See also Section 16.17.8.
   - Specify the cost of the reachable address: this must be greater than or equal to the circuit cost.
   - Specify the data format as PHASEIV, unless the receiving router is a DECNIS or another DECnet/OSI router (running the routing vector algorithm).

Figure 16–1 Interphase Links



Example

In Figure 16–1, the name of the circuit is Salt, and the Phase IV prefix for the network is 49. The router to which the DECNIS is connected is a Phase IV router. You would set up reachable addresses for Areas 9 and 10 as follows:

```
NCL> CREATE ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area09 -
_NCL> ADDRESS PREFIX 49::00-09

NCL> SET ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area09 COST 20, -
_NCL> TYPE OUTBOUND, DATA FORMAT PHASEIV

NCL> ENABLE ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area09

NCL> CREATE ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area0a -
_NCL> ADDRESS PREFIX 49::00-0A

NCL> SET ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area0a COST 20, -
_NCL> TYPE OUTBOUND, DATA FORMAT PHASEIV

NCL> ENABLE ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area0a
```

2. Create an inbound reachable address for each area in the link state domain.

Example

In Figure 16–1, you would set up a reachable address for Area 1 and Area 2 as follows:

```
NCL> CREATE ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area01 -
_NCL> ADDRESS PREFIX 49::00-01

NCL> SET ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area01 COST 20, -
_NCL> TYPE INBOUND

NCL> ENABLE ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area01

NCL> CREATE ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area02 -
_NCL> ADDRESS PREFIX 49::00-02

NCL> SET ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area02 COST 20, -
_NCL> TYPE INBOUND

NCL> ENABLE ROUTING CIRCUIT salt REACHABLE ADDRESS rtg$area02
```

16.17.8 Using Interphase Links to Connect to All Possible Phase IV Areas

To connect to all possible Phase IV areas, the DECNIS allows you to specify just four outbound reachable addresses, rather than 64.

The address prefixes that you must specify in this case (assuming a Phase IV prefix of 49::) are:

```
49::00-0
49::00-1
49::00-2
49::00-3
```

16.18 Using Dynamic Outbound Reachable Addresses

The reachable address attribute ADJACENCY DEPENDENT determines whether an outbound reachable address on a LAN circuit should be a **dynamic reachable address**.

A dynamic reachable address remains active only as long as the node specified in its LAN ADDRESS is up. If the LAN ADDRESS node goes down, the reachable address automatically becomes inactive; if the node recovers, the reachable address becomes active again.

16.18.1 Setting the ADJACENCY DEPENDENT Attribute

This section describes how to set the ADJACENCY DEPENDENT attribute.

16.18.1.1 Requirements

You can only set this attribute on CSMA/CD or FDDI circuits.

You can only set this attribute if the reachable address TYPE is OUTBOUND.

16.18.1.2 Procedure

To set the ADJACENCY DEPENDENT attribute, enter the following command:

```
NCL> SET ROUTING CIRCUIT circuit-name REACHABLE ADDRESS reachable-address -
_NCL> ADJACENCY DEPENDENT value
```

where *value* is either TRUE or FALSE.

Specify TRUE if you want the reachable address to be active or inactive according to whether the LAN ADDRESS node is up or down.

Specify FALSE if you want the reachable address to be permanently active, regardless of whether the LAN ADDRESS node is up or down.

16.18.2 Example

You want to create a reachable address for area 42, a Phase IV area. Area 42 is reachable through a Level 2 router in area 41, which is also a Phase IV area. The area 41 router has a LAN address of AA-00-04-00-23-A6.

You only want to use this reachable address if the area 41 router is up and running—that is, if the DECNIS on which you create the reachable address has a valid adjacency to the area 41 router.

To create the required dynamic reachable address, enter the following commands:

```
NCL> CREATE ROUTING CIRCUIT lan-0 REACHABLE ADDRESS area42 -
_ncl> ADDRESS PREFIX 49::00-2a

NCL> SET ROUTING CIRCUIT lan-0 REACHABLE ADDRESS area42 -
_NCL>LAN ADDRESS AA-00-04-00-23-A6 DATA FORMAT PHASEIV

NCL> SET ROUTING CIRCUIT lan-0 REACHABLE ADDRESS area42 -
_NCL> ADJACENCY DEPENDENT TRUE

NCL> ENABLE ROUTING CIRCUIT lan-0 REACHABLE ADDRESS area42
```

16.19 Connecting to LAN Systems Without Automatic Configuration

16.19.1 Introduction

You can configure a CSMA/CD or FDDI routing circuit to communicate with end systems that are ISO 8473 compliant, but do not support automatic configuration as defined by ISO 9542.

There are two methods of doing this, depending on whether or not the DECNIS is in the same area as the end system.

16.19.2 Connecting to End Systems in the Same Area as the DECNIS

In this case, you create an adjacency for the end system on the DECNIS, as follows:

Create an ADJACENCY entity for each end system with which you want to communicate. Specify the address and system-ID(s) of the LAN end system with which you want to communicate:

```
NCL> CREATE ROUTING CIRCUIT circuit-name ADJACENCY adjacency-name -
_NCL> LAN ADDRESS LAN-address, ENDNODE IDS {system-IDs}
```

where:

adjacency-name	is a name to identify the ADJACENCY entity. This name must be unique, not just for this circuit, but across all routing circuits.
system-IDs	are explained in Section 16.2.
LAN-address	is the LAN hardware address for the end system.

16.19.2.1 Example

The NSAP of the DECNIS is 37:1234:00-02:08-00-2B-65-43-21:00. The following command creates an adjacency to NODE_A, that has a LAN address of AA-00-04-00-01-08, and a system ID of 08-00-2B-12-34-56:

```
NCL> CREATE ROUTING CIRCUIT lan_circuit ADJACENCY node_a -
_NCL> LAN ADDRESS AA-00-04-00-01-08, ENDNODE IDS {08-00-2B-12-34-56}
```

The NSAP of the adjacency would be 37:1234:00-02:08-00-2B-12-34-56:00.

16.19.3 Connecting to End Systems in a Different Area from the DECNIS

In this case, you must create a reachable address for the end system on the DECNIS, as follows:

1. Disable the routing circuit:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit-name
   ```

2. Create a reachable address for each end system to be reached by this circuit. Specify an address prefix for each reachable address.

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name -
   _NCL> REACHABLE ADDRESS reachable-address ADDRESS PREFIX prefix
   ```

   where *reachable-address* is a name to identify the reachable address, and *prefix* is the full NSAP address of the end system.

3. Specify the LAN address to be used to connect with an address specified by the address prefix of the reachable address:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name
   _NCL> REACHABLE ADDRESS reachable-address LAN ADDRESS XX-XX-XX-XX-XX-XX
   ```

4. Enable the reachable address:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name -
   _NCL> REACHABLE ADDRESS reachable-address
   ```

5. Enable the circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

17
AppleTalk and IPX Routing

This chapter describes how to use the DECNIS for AppleTalk and NetWare IPX routing.

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

17.1 AppleTalk Routing

17.1.1 Introduction

The DECNIS software supports EtherTalk, Apple's specification for AppleTalk on Ethernet and FDDI. EtherTalk uses 802.2 SNAP format. AARP is supported for mapping between the AppleTalk Node Address and the Ethernet or FDDI hardware address.

AppleTalk is supported over wide area links using encapsulation of AppleTalk packets within IP packets.

17.1.2 AppleTalk Protocols Supported

The following AppleTalk protocols are supported by the DECNIS:

- AppleTalk Routing Table Maintenance Protocol (RMTP) which provides reachability information to other AppleTalk nodes (routers and end systems).
- AppleTalk Zone Information Protocol (ZIP) which provides mapping between AppleTalk networks and zones.
- AppleTalk Name Binding protocol (NBP) which provides numeric address mapping.
- AppleTalk Echo Protocol. The DECNIS can reply to an AppleTalk echo request as specified by the AppleTalk echo protocol.

17.1.3 Native and Tunnel Routing Circuits

There are two ways of using the DECNIS as an AppleTalk router:

1. By using native AppleTalk routing on CSMA/CD or FDDI circuits. These circuits route AppleTalk packets to and from AppleTalk printers, routers, servers and workstations.
2. By using tunnel circuits which encapsulate AppleTalk packets in Internet Protocol (IP) packets. Tunnel circuits can use synchronous lines, as well as CSMA/CD or FDDI circuits. Section 17.3 explains tunnel circuits.

17.1.4 Modes of AppleTalk Routing

The DECNIS can function as an AppleTalk local router, backbone router and/or half router. Note that in order for the DECNIS to function as a half router, you must use tunnel circuits.

If you decide not to use the DECNIS as an AppleTalk router, AppleTalk packets will be bridged by the DECNIS.

17.1.5 AppleTalk Addresses

An AppleTalk network is identified by a network address that is either a network number or a range of network numbers. A network number can be any number from 1 to 65,279. The number or range must be unique on the network; no two networks can have the same number and no two network ranges can overlap or have any network numbers in common.

Each number in a network range is a network address that can be associated with up to 253 nodes. The size of the network range determines the maximum number of nodes in the AppleTalk network.

Each AppleTalk node is identified by a node address. The address consists of two parts:

- The network number
- The node number which is also known as the node identifier (node ID)

Example: A node with the address 3.148 is for node ID 148 in network 3.

The AppleTalk address for the DECNIS is optional. If you use this address, the DECNIS will try to use it as the AppleTalk address for the DECNIS. Note the following:

- If you also use a network range, then the network number portion of this address must be within the network range.

- If the DECNIS finds that the node ID you have entered is already in use, it will dynamically assign another ID.

If you do not use this address, the DECNIS will generate its own AppleTalk address, as follows:

1. Network number. The DECNIS will find the network number:
   - From the network range, if you have entered it during configuration.
   - From the seed router, if you did not enter a range during configuration.
2. Node ID. The DECNIS will dynamically assign the node ID.

An AppleTalk node automatically receives an address by dynamic addressing when the node is started. This means that the address can change on system reboot if the old address has already been taken by another node.

17.1.5.1 AppleTalk Addresses and Tunnel Circuits

Do not associate an AppleTalk address with a tunnel circuit.

17.1.6 AppleTalk Name

The DECNIS does not register an AppleTalk name for itself. This means that the router will not be seen by various AppleTalk management tools, which rely on name registration to do autotopology (for example, node discovery).

17.1.7 Setting Up AppleTalk Routing on a LAN Routing Circuit

17.1.7.1 FDDI Restriction

On FDDI circuits, the NETWORK PROTOCOL of type APPLETALK must be created while the FDDI circuit is disabled. Therefore, on a running system, you must disable the FDDI routing circuit before step 1 in the following section.

17.1.7.2 Procedure

To configure a LAN routing circuit to use AppleTalk routing, follow these steps:

1. Ensure that a NETWORK PROTOCOL entity of type APPLETALK exists:

   ```
   NCL> CREATE ROUTING NETWORK PROTOCOL APPLETALK
   NCL> ENABLE ROUTING NETWORK PROTOCOL APPLETALK
   ```

   You only need to do this once, irrespective of the number of routing circuits on the DECNIS that will use AppleTalk routing.

2. Create a NETWORK PROTOCOL of type APPLETALK for the LAN routing circuit on which you want to use AppleTalk routing:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name -
   _NCL> NETWORK PROTOCOL APPLETALK
   ```

3. Specify the AppleTalk address for this routing circuit:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> NETWORK PROTOCOL APPLETALK -
   _NCL> APPLETALK MANUAL NETWORK ADDRESS {NETWORK = x, NODE = y}
   ```

   Section 17.1.5 describes the possible values of *x* and *y*.

4. Enable the NETWORK PROTOCOL entity for the circuit (note that the routing circuit must be enabled when you issue this command):

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name -
   _NCL> NETWORK PROTOCOL APPLETALK
   ```

17.1.7.3 Disabling AppleTalk Routing Circuits

To disable a routing circuit that is used for routing AppleTalk, you must first disable the AppleTalk network protocol for that circuit.

For example:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name NETWORK PROTOCOL -
_NCL> APPLETALK
```

```
NCL> DISABLE ROUTING CIRCUIT circuit-name
```

17.1.8 Seed Routers

In an AppleTalk network, the network range and zone list are defined on one of the routers. If this information is held on more than one router, they must all contain the same information.

The router that contains this information is known as the seed router. Every AppleTalk network contains at least one seed router. This router transmits the information to all other routers within the network.

If the DECNIS is a seed router on one of its routing circuits, you must specify a network range as follows:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL APPLETALK
```

```
NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL APPLETALK -
_NCL> APPLETALK MANUAL NETWORK RANGE [x..y]
```

```
NCL> ENABLE ROUTING CIRCUIT circuit-name NETWORK-
_NCL> PROTOCOL APPLETALK
```

x..y is a range of contiguous AppleTalk network numbers. The range specifies the range of network numbers available to nodes on the LAN to which the circuit connects. Each network range is unique to an AppleTalk network. Typically, each AppleTalk circuit connects to a different LAN, and therefore has a different network range.

If you enter a network range, then:

- You must also enter a default zone.
- The network number portion of the AppleTalk manual network address (if any) must be within the range.

17.1.8.1 Networks Without a Seed Router

If the DECNIS is not configured as a seed router for a particular network, and no other seed router is present on the network when the DECNIS starts, the DECNIS will not route AppleTalk packets.

To start the DECNIS routing AppleTalk packets, you must disable and reenable the AppleTalk network protocol on the circuit:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name NETWORK PROTOCOL -
_NCL> APPLETALK
```

```
NCL> ENABLE ROUTING CIRCUIT circuit-name NETWORK PROTOCOL -
_NCL> APPLETALK
```

17.1.9 AppleTalk Zones

An AppleTalk zone is a logical association of AppleTalk nodes within an Apple internet. The following information should be remembered when using zones:

- A zone has no physical definition—it can include one node, many nodes or all the nodes on the network.
- Nodes in the same zone do not have to be physically close, or on the same network or on neighboring networks.
- Zones are independent of network number.
- Any node in a network can belong in any zone whose name appears in that network’s zone list.
- A LocalTalk network can only be associated with one zone.

If your DECNIS is a seed router, use a a default AppleTalk zone.

If you use a default zone, you must also use a network range. The default zone is used:

- For AppleTalk nodes that cannot be preassigned with zone names.
- If an invalid zone name is assigned to a node.

You can use AppleTalk-specific characters in the zone name.

17.1.9.1 Setting a Default AppleTalk Zone

If the DECNIS is to act as a seed router on one of its routing circuits, set the default AppleTalk zone as follows:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL APPLETALK

NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL APPLETALK -
_NCL> APPLETALK PRIMARY ZONE zone-name

NCL> ENABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL APPLETALK
```

zone-name is the name of the default zone.

17.1.9.2 AppleTalk Zone List

If you wish, you may specify additional AppleTalk zones. These names are added to the seed zone list for the circuit. The seed zone list is a list of AppleTalk zone names that a router uses to establish the set of valid zone names for a network. You can use up to 31 additional AppleTalk zone names.

Specify additional AppleTalk zones as follows:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL APPLETALK

NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL APPLETALK -
_NCL> APPLETALK SECONDARY ZONES (zone-name1, zone-name2,...)

NCL> ENABLE ROUTING CIRCUIT circuit-name NETWORK-
_NCL> PROTOCOL APPLETALK
```

zone-name1 and *zone-name2* are the names of additional AppleTalk zones.

17.1.10 AppleTalk Characters

These are characters that do not exist in the DEC Multinational Character Set (DEC MCS), or are represented differently in the AppleTalk character set than in DEC MCS. Refer to your AppleTalk documentation for full details of the AppleTalk characters.

You can use character strings that contain AppleTalk specific characters for zone names. To do this, first find the AppleTalk character you want to use, and then enter the equivalent hexadecimal number, using the following format:

\ hexadecimal-number

You can use up to 8 backslashes, in addition to the 32 characters allowed for the zone name.

17.2 NetWare IPX Routing

17.2.1 Introduction

The DECNIS supports the routing of NetWare IPX packets. NetWare IPX packets can be routed to and from the following:

- NetWare IPX routers
- NetWare IPX workstations (clients)
- NetWare IPX servers (file and print servers)

17.2.2 NetWare Protocols Supported

The following NetWare protocols are supported by the DECNIS:

- Novell NetWare Routing Information Protocol (NetWare RIP), which provides the reachability of Novell NetWare routers and end systems.
- Novell NetWare Service Access Protocol (NetWare SAP), which provides information about the services available in the Novell NetWare network.

17.2.3 Native and Tunnel Routing Circuits

There are two ways to use the DECNIS as a NetWare IPX router:

1. By using native IPX routing, on the types of routing circuits listed in Section 17.2.4. These circuits route IPX packets to and from IPX end nodes.

2. By using tunnel circuits. These encapsulate IPX packets in Internet Protocol (IP) packets. Tunnel circuits can use any synchronous lines on which IP is configured. They can also use the same types of circuits as are used for native IPX routing. Section 17.3 explains tunnel circuits.

17.2.4 Routing Circuits for Native IPX Routing

You can run native NetWare IPX routing on the following types of routing circuits:

- CSMA/CD (see Section 9.3)
- FDDI (see Section 9.4)
- PPP, with either a PPP LINK or a CHDLC LINK specified for the Data Link Entity (see Section 9.9 and Section 9.11)
- ATM Permanent circuits.

17.2.5 NetWare Network Numbers

A NetWare network number identifies one of the following:

- A NetWare LAN.
- A NetWare point-to-point link.

Each NetWare LAN or point-to-point link must have a unique network number.

This number is assigned by the Novell administrator. It is an unused hexadecimal number between 1 and FFFFFFFF with a maximum length of 8 digits.

NetWare workstations learn their network number from NetWare servers or NetWare routers.

17.2.5.1 Network Numbers on LAN Circuits

On CSMA/CD and FDDI circuits, the network number identifies the LAN that the circuits connect to.

Giving Circuits the Same Network Number

You must give the LAN circuits the same network number if all the following are true:

- They connect to the same LAN.
- They are using the same data link encapsulation (framing).
- They are of the same circuit type, for example, two FDDI circuits.

Giving Circuits a Different Network Number

You must give the LAN circuits different network numbers if any of the following are true:

- They connect to different LANs.
- They have different data link encapsulation types.
- They are of different circuit types. For example, an FDDI circuit has a different network number from a CSMA/CD circuit.

17.2.5.2 Network Numbers on PPP, CHDLC and ATM Permanent Circuits

On PPP, CHDLC and ATM Permanent circuits, the network number identifies the point-to-point link. The network number must be the same at both ends of the link.

Each NetWare IPX link of type PPP or ATM on the DECNIS must have a different network number.

17.2.6 Setting Up Native NetWare IPX Routing

17.2.6.1 FDDI Restriction

On FDDI circuits, you can only create the NETWORK PROTOCOL of type NETWARE IPX while the FDDI circuit is disabled. Therefore, on a running system, you must disable the FDDI routing circuit before step 1 in Section 17.2.6.2.

17.2.6.2 Procedure

To configure a routing circuit to use NetWare IPX routing, follow these steps:

1. Ensure that a NETWORK PROTOCOL entity of type NETWARE IPX exists:

   ```
   NCL> CREATE ROUTING NETWORK PROTOCOL NETWARE IPX
   NCL> ENABLE ROUTING NETWORK PROTOCOL NETWARE IPX
   ```

   You only need to do this once, regardless of the number of routing circuits using NetWare IPX routing on the DECNIS.

2. Create a NETWORK PROTOCOL of type NETWARE IPX for the routing circuit on which you want to use NetWare IPX routing:

   ```
   NCL> CREATE ROUTING CIRCUIT circuit-name -
   _NCL> NETWORK PROTOCOL NETWARE IPX
   ```

3. Specify the NetWare address for this routing circuit:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name -
   _NCL> NETWORK PROTOCOL NETWARE IPX -
   _NCL> NETWARE NETWORK NUMBER %Xn
   ```

   Section 17.2.5 describes the possible values of *n*.

4. Enable the NETWORK PROTOCOL entity for the circuit (note that the routing circuit must be enabled when you issue this command):

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name -
   _NCL> NETWORK PROTOCOL NETWARE IPX
   ```

17.2.6.3 Disabling NetWare IPX Routing Circuits

To disable a routing circuit that is used for routing NetWare IPX, you must first disable the NetWare IPX network protocol for that circuit.

For example:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name NETWORK PROTOCOL NETWARE IPX
```

17.2.7 Frame Types on LAN Circuits

If you are running NetWare IPX on CSMA/CD or FDDI circuits, you must specify the frame type to be used by IPX on the circuit. You do not need to do this on PPP circuits.

Table 17–1 describes what frame type to use when using NetWare circuits on the DECNIS.

Table 17–1 Frame Types

Digital Term	Novell Term	When to Use
Ethernet	Ethernet II	DECnet, TCP/IP LAN, other LANs using Ethernet
802.2	Ethernet 802.2	LANs using the IEEE 802.2 standard
SNAP	Ethernet SNAP	LANs using the IEEE 802.2 SNAP extension
Novell	Ethernet 802.3	Networks using only NetWare

You can use all frame types on CSMA/CD connections, but only 802.2 and SNAP with FDDI connections.

17.2.7.1 Changing the Frame Type on a Routing Circuit

To set the frame type for IPX routing on a routing circuit, enter the following commands:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name NETWORK PROTOCOL NETWARE IPX
```

```
NCL> SET ROUTING CIRCUIT circuit-name NETWORK PROTOCOL NETWARE IPX -
_NCL> NETWARE DATA LINK ENCAPSULATION type
```

```
NCL> ENABLE ROUTING CIRCUIT circuit-name NETWORK PROTOCOL NETWARE IPX
```

17.2.8 Periodic Routing Protocol Timer

This is the number of seconds between periodic RIP and SAP updates on a circuit. The value is used by the DECNIS to determine how often the DECNIS should send NetWare RIP and SAP routing information updates on the circuit.

The value of this timer is a decimal number in the range 60–65535 seconds. Values are set in multiples of 60 seconds: the value you enter will be raised to the next multiple. For example, if you enter a value of 165, the timer will be given a value of 180 seconds. To disable regular updates, enter a value of 65535; updates will only be sent if the routing information has changed.

The value used for this timer is also used for ageing routers known by the circuit. Routers are aged (removed from the routing table for the circuit) if information about them is not broadcast within a time limit of 3 times the value of this timer.

17.2.8.1 Changing the Periodic Routing Protocol Timer

Change the value of this timer by issuing the following commands:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name NETWORK PROTOCOL NETWARE IPX
```

```
NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX -
_NCL> PERIODIC ROUTING PROTOCOL TIMER n
```

```
NCL> ENABLE ROUTING CIRCUIT circuit-name NETWORK PROTOCOL NETWARE IPX
```

17.2.9 NetBIOS Broadcast

NetBIOS™ is a protocol used by the IBM Token Ring/PC Network Interconnect Program to transmit messages between stations in a Token Ring network. NetBIOS runs over NetWare IPX circuits. If used, NetBIOS broadcast packets are sent as type 20 NetWare IPX packets which are forwarded on all circuits except for the circuit on which they originated.

17.2.9.1 Turning NetBIOS Broadcast On and Off

Issue the following commands to turn NetBIOS broadcast on or off:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX

NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX -
_NCL> NETWARE NETBIOS BROADCAST mode

NCL> ENABLE ROUTING CIRCUIT circuit-name NETWORK-
_NCL> PROTOCOL NETWARE IPX
```

where *mode* is ON or OFF.

17.3 Tunnel Circuits

17.3.1 Introduction

The DECNIS uses tunnel circuits to encapsulate NetWare IPX or AppleTalk packets in IP packets and route them to one or more destination IP addresses. You can use tunnel circuits on the DECNIS to:

- Forward PC LAN packets over synchronous lines.
- Forward a particular type of PC LAN packet to a router that does not support that PC LAN protocol.

Tunnel circuits are virtual circuits. This means that a tunnel circuit is not associated with a physical line, but only with a destination IP address, or addresses. On the DECNIS, tunnelling for NetWare IPX packets is implemented according to RFC 1234.

17.3.2 Maximum Number of Tunnel Circuits

You can have up to 32 NetWare IPX tunnel circuits, and up to 32 AppleTalk tunnel circuits on the DECNIS.

17.3.3 Types of Tunnel Circuit

There are two types of tunnel circuit:

- Point-to-point

  A point-to-point tunnel circuit has one destination IP address. You can send both NetWare IPX and AppleTalk packets on a single point-to-point tunnel circuit.

- Broadcast

  A broadcast tunnel circuit can have up to five destination IP addresses. You can only send NetWare IPX packets on a broadcast tunnel circuit.

17.3.4 Source Address for Tunnel Circuits

The source address used in IP packets sent on tunnel circuits is always the system IP address (also known as the manual IP address) for the DECNIS. Refer to Section 10.6 for more information about this address.

17.3.5 Setting Up a Point-to-Point Tunnel Circuit for AppleTalk or NetWare IPX Routing

Set up a point-to-point tunnel circuit as follows:

1. Create a tunnel circuit by issuing the following command:

```
NCL> CREATE ROUTING CIRCUIT circuit-name -
_NCL> TYPE = VIRTUAL POINT TO POINT
```

2. If you want to use this circuit for AppleTalk routing, create a NETWORK PROTOCOL entity of type APPLETALK; do not associate an AppleTalk address with tunnel circuits.

   If you want to use this circuit for NetWare IPX routing, create a NETWORK PROTOCOL entity of type NETWARE IPX, and set a NetWare IPX address, if required.

   If you want to use this circuit for both AppleTalk and IPX routing, create two NETWORK PROTOCOL entities, one of each type.

```
NCL> CREATE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL APPLETALK

NCL> CREATE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX

NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX
_NCL> NETWARE NETWORK NUMBER %Xn
```

3. Set the destination IP address for the circuit:

```
NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> ENCAPSULATION DESTINATION IP ADDRESS a.b.c.d
```

4. Enable the routing circuit and the NETWORK PROTOCOL entities:

```
NCL> ENABLE ROUTING CIRCUIT circuit-name
```

```
NCL> ENABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL APPLETALK
```

```
NCL> ENABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX
```

17.3.6 Setting Up a Broadcast Tunnel Circuit for NetWare IPX Routing

Set up a broadcast tunnel circuit as follows:

1. Create a tunnel circuit by issuing the following command:

```
NCL> CREATE ROUTING CIRCUIT circuit-name -
_NCL> TYPE = VIRTUAL BROADCAST
```

2. Create a NETWORK PROTOCOL entity of type NETWARE IPX.

```
NCL> CREATE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX
```

3. Set the NetWare address of the circuit:

```
NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX
_NCL> NETWARE NETWORK NUMBER %Xn
```

4. For each remote IP address for this circuit, create an adjacency and specify its IP address:

```
NCL> CREATE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX ADJACENCY adjacency-name
```

```
NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX ADJACENCY adjacency-name
_NCL> ENCAPSULATION IP ADDRESS a.b.c.d
```

5. Enable the routing circuit, the NETWORK PROTOCOL entity, and the adjacencies:

```
NCL> ENABLE ROUTING CIRCUIT circuit-name
```

```
NCL> ENABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX
```

```
NCL> ENABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX ADJACENCY adjacency-name
```

Part III

Managing X.25

This part contains information on managing the X.25 software on your system.

It contains the following chapters:

- Chapter 18 describes how to set up and manage DTEs, and use X.25 gateway functions of the DECNIS.
- Chapter 19 describes how to manage the X.25 relay functions of the DECNIS, explains how to set up and manage LLC2 DTEs, and describes how to use the DECNIS as a CONS LAN/WAN relay.
- Chapter 20 explains how to set up incoming and outgoing X.25 security on the DECNIS.

18

Managing X.25 Connections and X.25 Gateway Functions

This chapter describes how to manage DTEs and the X.25 gateway functions of the DECNIS. An X.25 gateway acts as a Connector system, allowing X.25 Client systems to access a PSDN (Packet Switched Data Network), or communicate across an X.25 point-to-point link.

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

Note

For any tasks that require you to disable a DTE, ensure that you do not disable the DTE that you are using to manage the DECNIS, unless there is an alternative route between the host on which you issue the NCL commands and the DECNIS.

18.1 Introduction

18.1.1 General

When using X.25 functions, the DECNIS has one or more connections to an X.25 network. An X.25 network can be a PSDN or a point-to-point link connecting two systems running X.25 software.

Each connection to an X.25 network requires one synchronous port on the DECNIS to be configured as a DTE. This DTE can also be used for other functions, for example, routing.

To configure the DECNIS as an X.25 gateway, you must create:

- DTEs
- DTE classes
- Server clients and associated filters
- Security entities (see Chapter 20)

18.1.2 Outgoing Calls

All the DTEs on the DECNIS are grouped into one or more DTE classes. All the DTEs in a DTE class should connect with the same X.25 network.

When a Client system wishes to communicate with a remote system, it sends a message to the DECNIS, specifying the DTE class to be used to make the call, the DTE address of the remote system with which it wishes to communicate, and various other call parameters.

The DECNIS then uses a DTE in the DTE class specified to send a call request packet to the remote system.

18.1.3 Incoming Calls

Each DECNIS has one or more filters. Each filter is associated with a Client system by means of an X25 ACCESS SERVER CLIENT entity.

When the DECNIS receives a call request from a remote DTE, it checks the parameters specified in the call request against parameters specified in its filters. When it finds a filter that matches the call request, it passes the call to the Client system associated with that filter.

18.2 Managing DTEs

18.2.1 Introduction

For every connection to a PSDN or X.25 point-to-point link, the DECNIS must have one synchronous hardware port configured as a DTE. Your subscription details for the PSDN will provide values for many of the DTE's characteristics.

18.2.2 Adding a DTE

1. Ensure that the MODEM CONNECT entity exists and is enabled by entering the following commands:

   ```
   NCL> CREATE MODEM CONNECT
   NCL> ENABLE MODEM CONNECT
   ```

A message will tell you if the MODEM CONNECT entity already exists; otherwise, it will be created.

2. Create a MODEM CONNECT LINE entity for the connection, specify the hardware port on which the connection will exist, and specify the profile to be used for the line.

   ```
   NCL> CREATE MODEM CONNECT LINE line-name -
   _NCL> COMMUNICATION PORT port-name, PROFILE line-profile
   ```

   where *line-profile* is either "NORMAL" or "DATEXP". The profile specifies the default value and permitted range for certain timers on the line. See the NCL online help for details of these timers.

3. Set the MODEM CONTROL characteristic for the line:

   ```
   NCL> SET MODEM CONNECT LINE line-name MODEM CONTROL control
   ```

   where *control* is either FULL or NONE. FULL is the normal setting, and means that the line takes note of the modem control signals. Set it to NONE if the line is to ignore all modem control signals (for example, during loopback testing).

4. Enable the line:

   ```
   NCL> ENABLE MODEM CONNECT LINE line-name
   ```

5. Ensure that the LAPB entity exists:

   ```
   NCL> CREATE LAPB
   ```

6. Create a LAPB data link for the circuit, and specify the profile that the link will use:

   ```
   NCL> CREATE LAPB LINK link-name PROFILE profile-name
   ```

   where:

   link-name is a name identifying the data link. You should make this the same as the communication port name.

   profile-name is the name of the profile to use for this link. The name is determined by the PSDN to which the circuit will connect: the online *Network Information* provides the profile name for each public PSDN that the DECNIS supports.

7. Set the acknowledge timer for the link: see Section 18.2.4.

8. Set the holdback timer for the link: see Section 18.2.5.

9. Set the window size for the link: see Section 18.2.6.

10. Set the maximum frame size for the link: see Section 18.2.7.

11. Associate the LINE created in step 2 with this data link:

    ```
    NCL> SET LAPB LINK link-name -
    _NCL> PHYSICAL LINE MODEM CONNECT LINE line-name
    ```

12. Enable the LAPB data link:

    ```
    NCL> ENABLE LAPB LINK link-name
    ```

13. Set up a MOP circuit on the data link, to allow subsequent loopback testing: see Section 8.26.

14. Ensure that the X25 PROTOCOL entity exists:

    ```
    NCL> CREATE X25 PROTOCOL
    ```

15. Create the DTE entity and specify the profile it will use:

    ```
    NCL> CREATE X25 PROTOCOL DTE DTE-name PROFILE "profile-name"
    ```

    where *profile-name* is the name of the profile that this DTE will use. The name is determined by the PSDN to which the circuit will connect, or the type of point-to-point link. *Network Information* provides the profile name for each public PSDN that the DECNIS supports.

    The profile you enter here must be the one you entered when you created the LAPB link in step 5.

    Note that you must include the quotation marks around the profile name.

16. Specify that the DTE is to use the LAPB link set up previously:

    ```
    NCL> SET X25 PROTOCOL DTE DTE-name -
    _NCL> LINK SERVICE PROVIDER LAPB LINK link-name
    ```

17. Specify the DTE address of this DTE (provided by the PSDN):

    ```
    NCL> SET X25 PROTOCOL DTE DTE-name X25 ADDRESS DTE-address
    ```

    where *DTE-address* is up to 15 decimal digits.

18. Specify the DTE class with which calls received on this DTE will be associated.

    ```
    NCL> SET X25 PROTOCOL DTE DTE-name INBOUND DTE CLASS DTE-class
    ```

    Usually, you should make *DTE-class* the same as the DTE class to which this DTE belongs (see Section 18.3). Note that, if it does not already exist, you must create a DTE class with this name: see Section 18.3.2.

19. Allocate channels for incoming and outgoing calls, according to the PSDN subscription information. For example, assume that a PSDN has allocated the following channels for use with this DTE:

    - 1 to 32 both-way
    - 49 to 64 outgoing only
    - 65 to 512 incoming only

    Specify those channels that are to be used for outgoing calls; include both-way channels:

    ```
    NCL> SET X25 PROTOCOL DTE DTE-name OUTGOING LIST [[1..32],[49..64]]
    ```

    Now specify those channels that are to be used for incoming calls; again, include both-way channels:

    ```
    NCL> SET X25 PROTOCOL DTE DTE-name INCOMING LIST [[1..32],[65..512]]
    ```

    Note that you must not specify any channels here that are to be used for PVCs (see Section 18.6).

20. If this DTE is to be used for an X.25 point-to-point link, you must choose which of the two systems is to act as the DTE, and which as the DCE: see Section 18.2.10.1.

21. Add the DTE to a DTE class on your system: see Section 18.3.4. This will usually be the DTE class specified in step 17.

22. Enable the DTE:

    ```
    NCL> ENABLE X25 PROTOCOL DTE DTE-name
    ```

18.2.3 Disable Loading/Dumping Over X.25 Lines

If the DECNIS cannot load over its normal load circuit it tries all its other connections. However, lines connected to X.25 networks cannot be used for loading or dumping. You must prevent the DECNIS from attempting to load over such lines by disabling loading and dumping on any Network Interface Card on which a DTE has beem set up. *DECNIS Configuration and Loading* explains how to prevent lines associated with a Network Interface Card from attempting to load or dump.

18.2.4 Changing the Acknowledge Timer

After sending a message, a data link will wait for a time determined by the acknowledge timer. If the data link does not receive a response within this time, it will retransmit the message.

Decrease this value from the default (3000) if you are using a fast or noisy link. Increase this value if you are using a slow link.

The value of the acknowledge timer should be set as follows:

```
Acknowledge Timer >= (2 * maxfrm) + Holdback Timer
```

where `maxfrm` is the time taken for a maximum-sized frame to be transmitted by the DECNIS and received at the adjacent system, and `Holdback Timer` is the value of the holdback timer at the adjacent system.

Set the ACKNOWLEDGE TIMER characteristic of the data link:

```
NCL> SET LAPB LINK link-name ACKNOWLEDGE TIMER n
```

where *n* is a decimal number between 1 and 60,000, and specifies the time in milliseconds.

Note that when you change the value of this timer on a running system, the timer is reset.

18.2.5 Changing the Holdback Timer

The HOLDBACK TIMER characteristic specifies the delay (in milliseconds) before an explicit acknowledgment must be sent if there are no data frames available to carry an implicit acknowledgment.

The value of the holdback timer is related to the value of the acknowledge timer for the link as described in Section 18.2.4.

Set the HOLDBACK TIMER characteristeric of the data link:

```
NCL> SET LAPB LINK link-name HOLDBACK TIMER n
```

where *n* is a decimal number between 0 and 60,000 and specifies the time in milliseconds.

Note that when you change the value of this timer on a running system, the timer is reset.

18.2.6 Changing the Window Size

You can change the frame-level window size for this circuit. Increase the value from the default (which depends on the DTE profile) if there is a significant delay on the link.

However, note that the value of the frame-level window size must be the same at the DTE and the DCE, and so must not be changed without the agreement of the PSDN authority.

Set the WINDOW SIZE characteristic of the data link:

```
NCL> SET LAPB LINK link-name WINDOW SIZE n
```

where *n* is a decimal number between 1 and 127. Note that if you set the WINDOW SIZE characteristic to a value greater than 7, you must set the SEQUENCE MODULUS characteristic to 128.

18.2.7 Changing the Maximum Frame Size

You can change the maximum frame size to be used by the LAPB data link.

However, note that the value of the maximum frame size must be the same at the DTE and the DCE, and so must not be changed without the agreement of the PSDN authority.

Set the MAXIMUM DATA SIZE characteristic of the data link:

```
NCL> SET LAPB LINK link-name MAXIMUM DATA SIZE n
```

where *n* is a decimal number between 1 and 65,532, and specifies the frame size (including the frame header) in bytes.

18.2.8 Monitoring a DTE

You can use the following commands to check how a DTE is functioning:

- To check the state of a DTE:

  ```
  NCL> SHOW X25 PROTOCOL DTE DTE-name STATE
  ```

  This shows the state of the DTE. See the NCL online help for a description of the states displayed.

- To check how a DTE is being used:

  ```
  NCL> SHOW X25 PROTOCOL DTE DTE-name ALL COUNTERS
  ```

  The counters will increment only when calls are attempted or accepted using this DTE.

  See the NCL online help for a description of the counters displayed.

18.2.8.1 X.25 Accounting

The X25 ACCESS entity generates an event called PORT TERMINATED. The arguments to this event provide basic accounting information, such as DTE class, call direction, and so on. (See the NCL online help for a full list of the arguments and their meanings.)

This information can be processed by a user-written application to provide X.25 accounting information, similar to that provided by the VAX™ P.S.I. product.

18.2.9 Deleting a DTE

Delete a DTE if you want to prevent permanently the DECNIS from communicating with the network through that DTE. Use the DISABLE command if you want to prevent temporarily the DECNIS from communicating with the network through the DTE.

1. Delete any PVCs associated with this DTE:

   ```
   NCL> DELETE X25 PROTOCOL DTE DTE-name PVC *
   ```

2. Disable and delete the DTE:

   ```
   NCL> DISABLE X25 PROTOCOL DTE DTE-name
   NCL> DELETE X25 PROTOCOL DTE DTE-name
   ```

3. Amend any groups of which this DTE was a member: see Section 18.7.4.
4. Amend the list of members of the DTE class of which this DTE was a member: see Section 18.3.4.
5. Delete the DTE class, if it is now empty: see Section 18.3.3.
6. Disable and delete the LAPB link associated with this DTE:

   ```
   NCL> DISABLE LAPB LINK link-name
   NCL> DELETE LAPB LINK link-name
   ```

18.2.10 Changing DTE Characteristics

The default values and permitted ranges for several DTE characteristics are determined from the profile that the DTE is using. In addition, if the DTE connects to a PSDN, the subscription details for the DTE will determine the values you can give to some characteristics. For example, the profile may permit a range of values for the DEFAULT PACKET SIZE characteristic, but your subscription options for this DTE may disallow any value other than the profile default.

The NCL online help lists all the characteristics of the DTE entity, and explains how to change them.

18.2.10.1 Point-to-Point Links: DTE Interface Type

If the X.25 network to which the DTE connects is a point-to-point link, one system must act as the DTE, and the other as the DCE. This section describes how to change the interface type.

The default interface type is DTE; make your system the DCE as follows:

1. Disable the DTE, and change the interface type:

   ```
   NCL> DISABLE X25 PROTOCOL DTE DTE-name
   NCL> SET X25 PROTOCOL DTE DTE-name INTERFACE TYPE DCE
   NCL> ENABLE X25 PROTOCOL DTE DTE-name
   ```

2. If the interface type of the DTE is DCE, you should also operate the DECNIS as a DCE at the link level (provided that the remote system is operating as a DTE at the link level):

   ```
   NCL> DISABLE LAPB LINK link-name
   NCL> SET LAPB LINK link-name INTERFACE TYPE DCE
   NCL> ENABLE LAPB LINK link-name
   ```

18.2.10.2 Packet and Window Size

You can alter the various packet and window size characteristics of the DTE to take account of your particular circumstances, provided you have the appropriate PSDN subscription options for the DTE.

Packet Size

The packet size used for a call determines the amount of data that can be sent in each packet. You should consider using a larger packet size if you have large amounts of data to send.

There are three packet size characteristics:

- MAXIMUM PACKET SIZE
- DEFAULT PACKET SIZE
- MINIMUM PACKET SIZE

If the PSDN subscription for the DTE does not allow packet-size negotiation, all calls made using the DTE use the value specified by DEFAULT PACKET SIZE. If the DTE subscribes to, and the call is using, packet-size negotiation, the packet size used for a call on this DTE will be somewhere between the value of the MAXIMUM PACKET SIZE and that of the MINIMUM PACKET SIZE.

The value for DEFAULT PACKET SIZE must be less than or equal to that for MAXIMUM PACKET SIZE, and greater than or equal to that for MINIMUM PACKET SIZE.

Note that the value for DEFAULT PACKET SIZE must match the value used by the DCE. Therefore, do not change this value without the agreement of the PSDN authority.

Window Size

The window size used for a call determines the maximum number of packets that a DTE can send to a remote DTE without waiting for an acknowledgment. When the DTE reaches this limit, it must wait for an acknowledgment from the remote DTE before it can transmit any further packets.

You should consider using a larger window size, in conjunction with extended packet sequence numbering, if you have large amounts of data to send, or if there is a large propagation delay in your network (for example, if it includes a satellite link).

There are four characteristics related to window size:

- MAXIMUM WINDOW SIZE
- DEFAULT WINDOW SIZE
- MINIMUM WINDOW SIZE
- EXTENDED PACKET SEQUENCING

If the PSDN subscription for the DTE does not allow window-size negotiation, all calls made using the DTE use the value specified by DEFAULT WINDOW SIZE. If the DTE subscribes to, and the call is using, window-size negotiation, the window size used for a call on this DTE will be somewhere between the value of the MAXIMUM WINDOW SIZE and that of the MINIMUM WINDOW SIZE.

The value for DEFAULT WINDOW SIZE must be less than or equal to that for MAXIMUM WINDOW SIZE, and greater than or equal to that for MINIMUM WINDOW SIZE.

Note that the value for DEFAULT WINDOW SIZE must match the value used by the DCE. Therefore, do not change this value without the agreement of the PSDN authority.

Set the EXTENDED PACKET SEQUENCING characteristic to TRUE if you want sequence numbering of packets to be modulo 128; set it to FALSE if you want it to be modulo 8.

Note that you can only set this characteristic to TRUE if the profile you specified for the DTE supports extend packet sequence numbering, for example ISO8208_E.

18.3 Managing DTE Classes

18.3.1 Managing DTE Classes

Each DTE on the DECNIS must be in a DTE class. A DTE class can have several DTEs, and a DTE can be in more than one DTE class.

All DTEs in a DTE class should be connected to the same X.25 network.

When a call is received at the DECNIS, it acquires an INBOUND DTE CLASS attribute from the DTE at which it was received. Normally this should be the same as the DTE class to which the DTE belongs.

18.3.2 Adding a DTE Class

1. Ensure that the X25 ACCESS entity exists:

   ```
   NCL> CREATE X25 ACCESS
   ```

2. Create the DTE class, specifying that it is of type LOCAL:

   ```
   NCL> CREATE X25 ACCESS DTE CLASS class-name TYPE LOCAL
   ```

3. Add the DTEs that this class is to contain: see Section 18.3.4.

4. Set up security for this DTE class: see Chapter 20.

18.3.3 Deleting a DTE Class

Delete a DTE class if you want to prevent permanently any Client systems from using it to make outgoing calls. The Client systems should also delete their corresponding DTE class. Note that you cannot disable a DTE class.

1. Disable or delete the DTEs that are members of the DTE class (see Section 18.2.9), or put them in another DTE class (see Section 18.3.4).

2. Ensure that no enabled DTEs specify the DTE class as their INBOUND DTE CLASS characteristic (see Section 18.2.2).

3. When the DTE class is empty, or all its member DTEs are disabled, delete the DTE CLASS entity:

   ```
   NCL> DELETE X25 ACCESS DTE CLASS class-name
   ```

18.3.4 Changing Members of a DTE Class

You can change the list of DTEs in a DTE class.

1. Add a DTE to the DTE class as follows:

   ```
   NCL> ADD X25 ACCESS DTE CLASS class-name LOCAL DTES {DTE-name}
   ```

2. Remove a DTE from a DTE class as follows:

   ```
   NCL> REMOVE X25 ACCESS DTE CLASS class-name LOCAL DTES {DTE-name}
   ```

   If the DTE you wish to remove is currently enabled, you must disable it before you remove it:

   ```
   NCL> DISABLE X25 PROTOCOL DTE DTE-name
   ```

   Ensure that you do not remove all the DTEs from a DTE class. If you do, then you should delete that DTE class.

3. Change the INBOUND DTE CLASS attribute of any DTEs that have moved from one DTE class to another: see Section 18.2.2.

4. Finally, make the DTE available through the DTE class by disabling and reenabling the DTE:

   ```
   NCL> DISABLE X25 PROTOCOL DTE DTE-name
   NCL> ENABLE X25 PROTOCOL DTE DTE-name
   ```

18.4 Managing Server Clients

18.4.1 Introduction

A server client defines a Client system that uses the DECNIS as a Connector system to receive calls. Each server client has one or more filters associated with it (see Section 18.5).

When a call is received at the DECNIS, it is checked against the filters on the system. The highest priority filter whose characteristics match the call characteristics is noted, and the call is passed to the Client system defined by the server client associated with that filter. Figure 18–1 illustrates the operation of filters and server clients on the DECNIS.

18.4.2 Adding a Server Client

1. Ensure that the X25 SERVER entity exists:

   ```
   NCL> CREATE X25 SERVER
   ```

2. Create a CLIENT entity for the Client system:

   ```
   NCL> CREATE X25 SERVER CLIENT client-name
   ```

3. Specify the filter(s) that are to be used to select calls for the CLIENT entity:

   ```
   NCL> SET X25 SERVER CLIENT client-name FILTERS {filter-name}
   ```

   where *filter-name* is the name of the filter to be used for this server client. If you specify more than one filter name, separate them with a comma.

   You must associate each filter with only one server client.

   The filters specified do not need to exist before you create the server client. However, the server client cannot be enabled until its filters exist.

4. Specify the Client system that is to be associated with the server client: see Section 18.4.4.

5. Create the filters specified above, if necessary: see Section 18.5.2.

6. Enable the CLIENT entity:

   ```
   NCL> ENABLE X25 SERVER CLIENT client-name
   ```

18.4.3 Deleting a Server Client

Delete a server client if you want to prevent permanently any calls from reaching the system specified in it. Use the DISABLE command if you want to prevent temporarily any calls from reaching the system specified in the server client.

1. Disable and delete the CLIENT entity:

   ```
   NCL> DISABLE X25 SERVER CLIENT client-name
   NCL> DELETE X25 SERVER CLIENT client-name
   ```

2. Now delete the filter(s) that were associated with the server client: see Section 18.5.3.

18.4.4 Changing the Client System Associated with a Server Client

Change the Client system associated with the server client using the following command:

```
NCL> SET X25 SERVER CLIENT client-name -
_NCL> NODE node-id
```

where *node-id* is the node name of the Client system to be associated with the server client, as defined in the Known Towers database (see Section 1.11).

18.4.5 Changing the List of Filters Associated with a Server Client

1. Disable the CLIENT entity:

   ```
   NCL> DISABLE X25 SERVER CLIENT client-name
   ```

2. Add a filter to the server client as follows:

   ```
   NCL> ADD X25 SERVER CLIENT client-name FILTERS {filter-name}
   ```

3. Remove a filter from the server client as follows:

   ```
   NCL> REMOVE X25 SERVER CLIENT client-name FILTERS {filter-name}
   ```

   Note that removing a filter will have no effect on current calls that were established using that filter.

4. Enable the CLIENT entity:

   ```
   NCL> ENABLE X25 SERVER CLIENT client-name
   ```

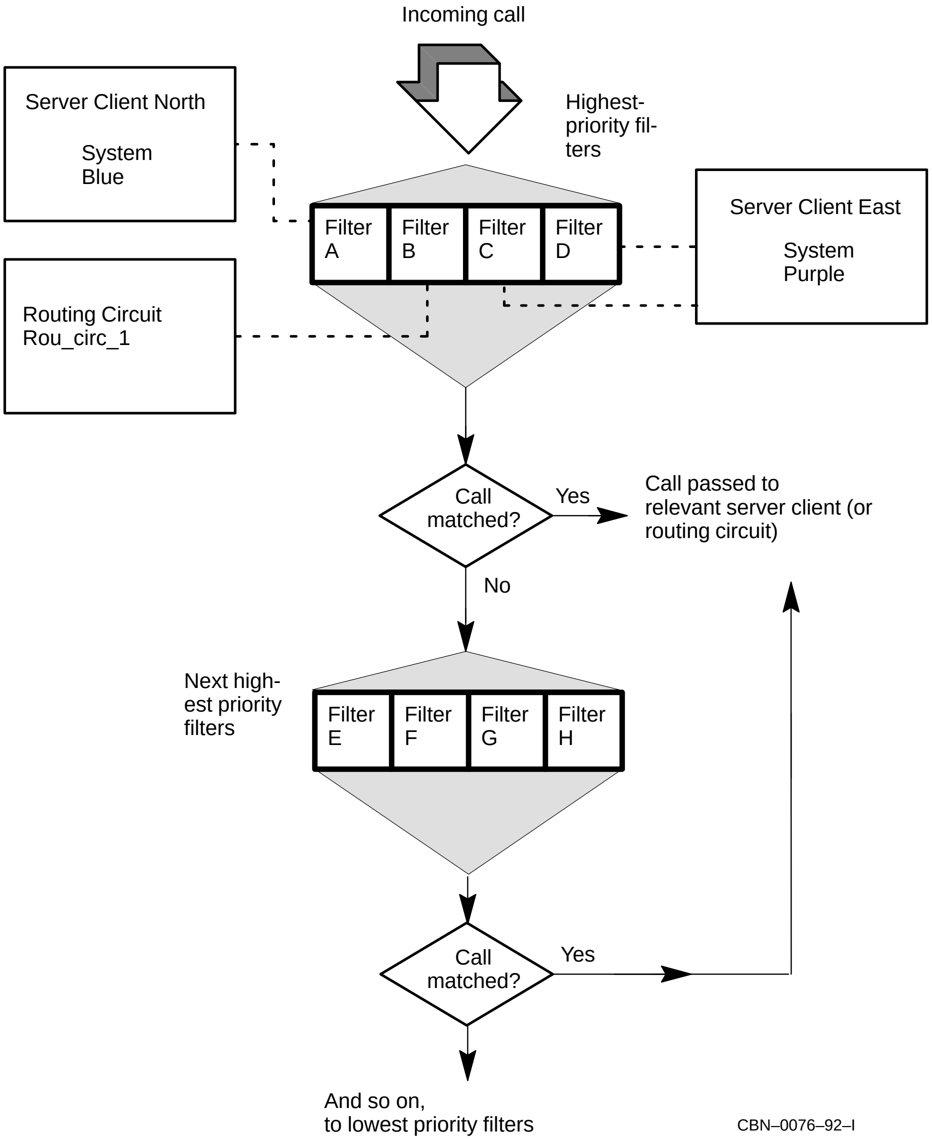
18.5 Managing Filters

18.5.1 Introduction

Filters are used on the DECNIS to allocate incoming calls. For X.25 gateway functions, filters are associated with server clients. If the DECNIS has routing circuits of type X.25 DA or X.25 STATIC INCOMING, then at least one filter must be associated with each of these routing circuits, so that calls received on these circuits can be correctly dealt with: see Section 9.16. Figure 18–1 describes the operation of filters on the DECNIS.

An incoming call is checked first against all the highest-priority filters. If a match is found, the call is passed to the Client system defined by the server client with which the filter is associated (or to a routing circuit, if the filter is associated with that). For example, in Figure 18–1, if Filter C matches the incoming call, the call will be passed to System Purple. If Filter B matches the incoming call, the call will be passed to routing circuit Rou_circ_1.

Figure 18–1 How Filters Handle Incoming Calls on the DECNIS



If a match is not found, the call is checked against the next-highest priority filters, and the process is repeated until a match is found.

If no matching filter is found at any priority, the call is rejected and an event is logged.

Filters have characteristics that correspond with parameters in call request packets. A filter matches a call request when each characteristic in the filter is either the same as the corresponding parameter in the call request packet, or is unspecified.

Each filter has a priority. As shown in Figure 18–1, higher-priority filters are checked before lower-priority filters.

It is possible to create a catchall filter, which is a low-priority filter that has most of its characteristics unspecified. In this way, it would match nearly all incoming calls, but would only be used when all higher-priority filters had been tried.

18.5.2 Adding a Filter

1. Ensure that the X25 ACCESS entity exists:

   ```
   NCL> CREATE X25 ACCESS
   ```

2. Create a FILTER entity:

   ```
   NCL> CREATE X25 ACCESS FILTER filter-name
   ```

3. Set the filter characteristics that you require: see Section 18.5.4.
4. Set up security for the filter: see Chapter 20.

18.5.3 Deleting a Filter

Delete a filter if you want to prevent permanently any calls from matching with it. Note that you cannot disable a filter, although you can disable the server client or routing circuit that is using it.

1. Amend the list of filters associated with the server client: see Section 18.4.5.
2. Delete the FILTER entity:

   ```
   NCL> DELETE X25 ACCESS FILTER filter-name
   ```

18.5.4 Changing Filter Characteristics

The characteristics set in the filter are used to check the corresponding parameters in a call request packet.

The NCL online help lists all the characteristics of the filter entity, and explains how to change them.

You can leave all the characteristics except PRIORITY and SECURITY FILTER unspecified. The PRIORITY characteristic is a decimal number between 0 and 65,535; the lower the number, the lower the priority. The SECURITY FILTER characteristic specifies the name of the security filter to be used by the filter. The default is "default".

18.6 Managing PVCs

18.6.1 Introduction

A PVC (Permanent Virtual Circuit) is a permanent association between two systems, across an X.25 network, so that the systems can communicate without any need for call setup or call clearing.

You can set up one or more PVCs on each DTE. Your PSDN subscription information will specify details of any PVCs you can set up.

Note that PVCs on the DECNIS must connect to PVCs on the remote system. Connecting a PVC to an SVC will cause incorrect operation.

18.6.2 Adding a PVC

1. Create the PVC entity, and specify the channel over which it will operate, and the packet and window sizes it will use:

   ```
   NCL> CREATE X25 PROTOCOL DTE DTE-name PVC PVC-name CHANNEL channel,-
   _NCL> PACKET SIZE packet-size, WINDOW SIZE window-size
   ```

   where:

   channel is a decimal number between 1 and 4095, and is supplied by the PSDN. The number you specify here must not be part of the INCOMING LIST or OUTGOING LIST specified in Section 18.2.2.

   packet-size is a decimal number that is a power of 2 between 16 and 4096. The value you enter is supplied by the PSDN.

   window-size is a decimal number between 1 and 127. The value you enter is supplied by the PSDN.

2. Set up security for the PVC: see Chapter 20.

18.6.3 Deleting a PVC

Delete a PVC if you want to prevent permanently the DECNIS using that PVC to communicate with the remote DTE. Note that you cannot disable a PVC.

Delete the PVC entity as follows:

```
NCL> DELETE X25 PROTOCOL DTE DTE-name PVC PVC-name
```

18.7 Managing Groups

18.7.1 Introduction

If a DTE belongs to a Closed User Group (CUG), it can communicate freely with remote DTEs that are also members of that group. However, depending on the PSDN subscription options, communication with remote DTEs outside the CUG may be restricted.

The PSDN subscription will give details of the CUG(s) to which each DTE can belong.

18.7.2 Adding a Group

1. Create the GROUP entity:

   ```
   NCL> CREATE X25 PROTOCOL GROUP group-name
   ```

   where *group-name* is a name used by the DECNIS to identify the group.

2. Specify the type of group:

   ```
   NCL> SET X25 PROTOCOL GROUP group-name TYPE group-type
   ```

   where *group-type* is either CUG or BCUG:

   - BCUG indicates a Bilateral Closed User Group, that is, one that has just one local DTE and one remote DTE.
   - CUG indicates a Closed User Group that can have more than two members.

3. Enter the local DTE(s) that are to be members of this group: see Section 18.7.4.

4. If the group type is BCUG, specify the DTE address of the remote member of the BCUG:

   ```
   NCL> SET X25 PROTOCOL GROUP group-name -
   _NCL> REMOTE DTE ADDRESS DTE-address
   ```

   where *DTE-address* is the DTE address of the other member of the BCUG.

18.7.3 Deleting a Group

Delete a group if you want to prevent permanently the DECNIS from communicating with the network through that group. Note that you cannot disable groups.

1. If the group contains local DTEs that are currently enabled, you must disable them first:

   ```
   NCL> DISABLE X25 PROTOCOL DTE DTE-name
   ```

2. Delete the GROUP entity:

   ```
   NCL> DELETE X25 PROTOCOL GROUP group-name
   ```

18.7.4 Changing Members of a Group

This section describes how to add local DTEs to a group, and remove local DTEs from a group.

1. Add local DTE(s) to a group by specifying the name(s) of the DTE(s) to be added, together with the Group Number supplied by the PSDN for this DTE/Group:

   ```
   NCL> ADD X25 PROTOCOL GROUP group-name MEMBERS = -
   _NCL> {(DTE=DTE-name_1, INDEX=n1), (DTE=DTE-name_2, INDEX=n2), ...}
   ```

   where:

DTE-name_1, *DTE-name_2*	are the names of the DTEs on the DECNIS.
n1, *n2*	are decimal numbers, supplied by the PSDN, between 1 and 9999.

2. Remove local DTE(s) from this group by specifying the name(s) of the DTE(s) to be removed, together with the Group Number supplied by the PSDN for this DTE/Group:

   ```
   NCL> REMOVE X25 PROTOCOL GROUP group-name -
   _NCL> MEMBERS = {(DTE=DTE-name_1, INDEX=n1)}
   ```

Note that you do not specify the remote DTEs that are members of the group: the PSDN deals with this information.

19

X.25 Relay

19.1 Introduction

This chapter describes how to manage the X.25 relay functions of the DECNIS. It describes what an X.25 relay is and how to set up relay functions. It also describes how to set up LLC2 DTEs, and use the relay functions to operate the DECNIS as a CONS LAN/WAN relay.

19.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

Note

For any tasks that require you to disable a DTE, ensure that you do not disable the DTE that you are using to manage the DECNIS, unless there is an alternative route between the host on which you issue the NCL commands and the DECNIS.

19.2 Description of the X.25 Relay

The X.25 relay functions of the DECNIS allow host systems to communicate by switching X.25 calls between them. Incoming calls can be switched to another DTE within the DECNIS, or to a DTE on another X.25 relay. In addition, the DECNIS can use its relay functions to act as a CONS LAN/WAN relay (see Section 19.8).

Figure 19–1 Principles of X.25 Relay

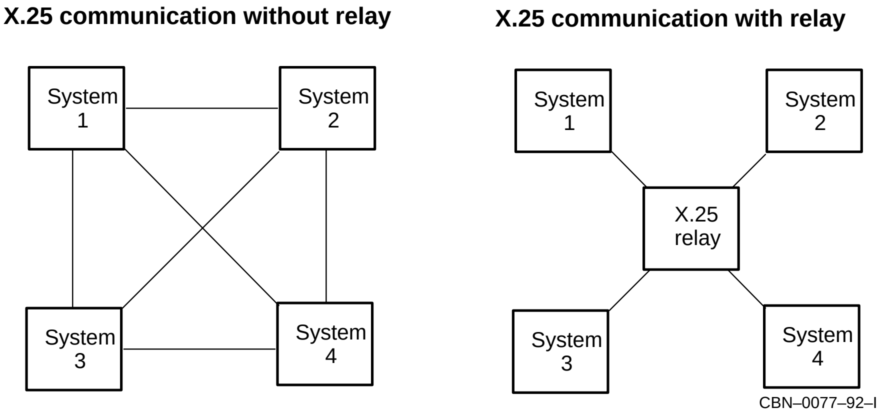
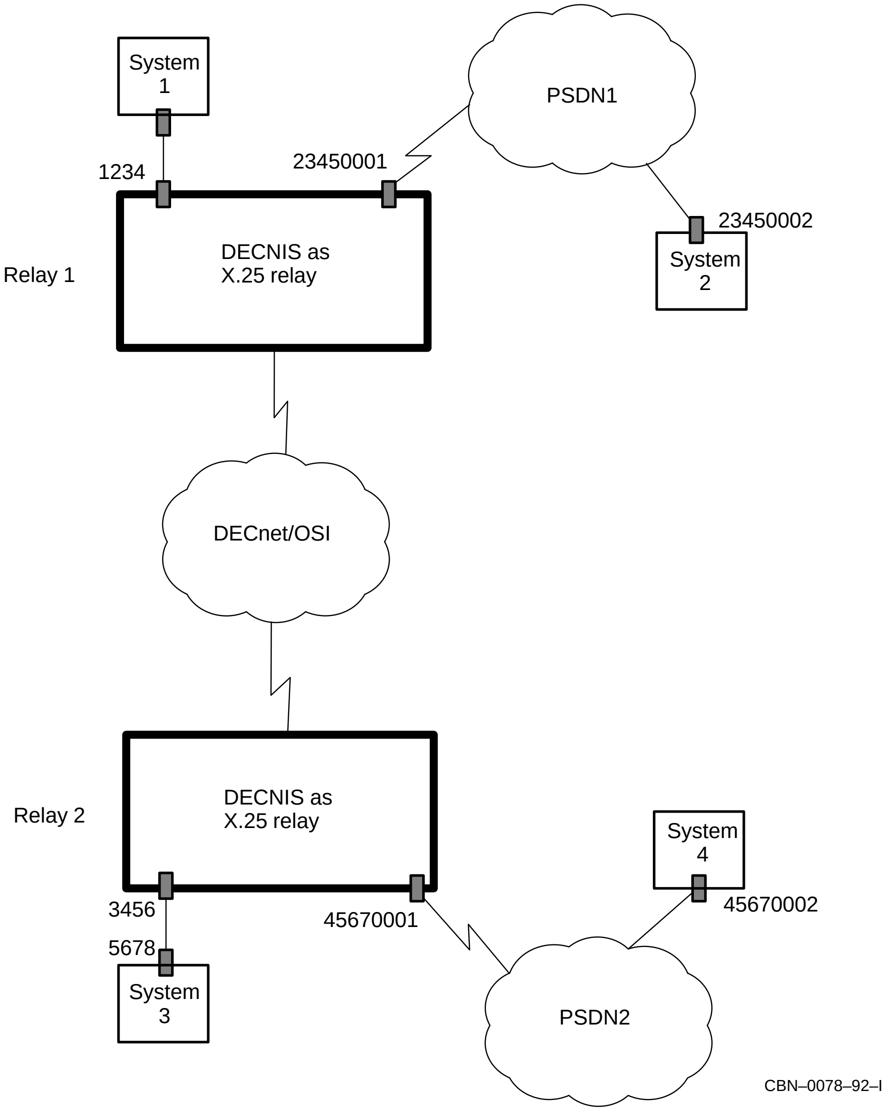


Figure 19–1 illustrates the principle of X.25 relay operation. Systems 1, 2, 3, and 4 wish to communicate using X.25 software. In the left diagram, each system sets up a point-to-point X.25 link with every other system. In the right diagram, the same connectivity is achieved, using fewer lines and DTEs, by using an X.25 relay. Each system has just one point-to-point link, and the relay switches calls between them as necessary. Note that Systems 1, 2, 3, and 4 need not be Digital systems.

Figure 19–2 shows a more realistic example of the use of an X.25 relay. In this figure, all host systems can communicate, except Systems 2 and 4: the PSDN authorities only allow communication between PSDNs by use of an X.75 link.

Figure 19–2 DECNIS Performing More Complex Relay Functions



19.3 How the DECNIS Operates as an X.25 Relay

19.3.1 Introduction

An X.25 relay connects to X.25 networks using DTEs; Section 18.2.2 explains how to set up DTEs.

Each X.25 network to which calls are to be switched is known as a relay client of the DECNIS, and is defined on the DECNIS by a RELAY CLIENT entity.

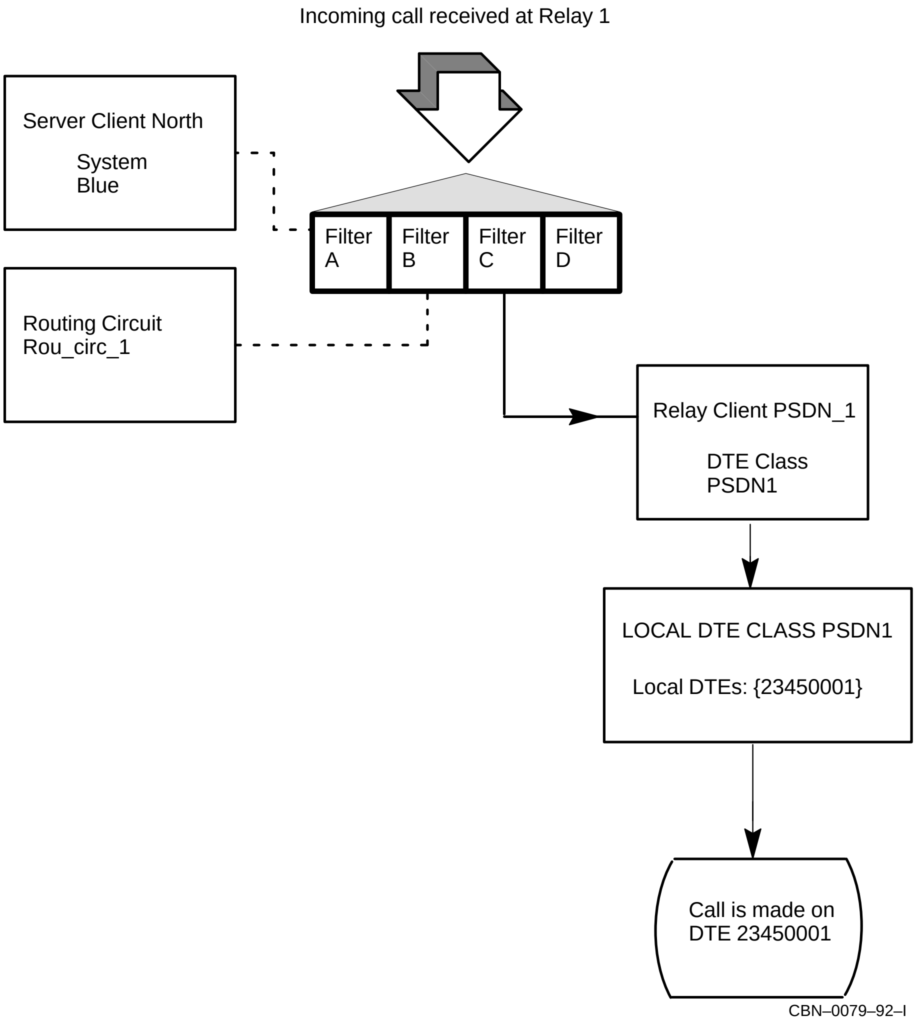
For example, for System 1 to make an X.25 call to System 2, the following occurs (see Figure 19–3):

1. System 1 makes an X.25 call to Relay 1, using its point-to-point link to Relay 1's DTE 1234. The destination DTE address specified in this call is 23450002.
2. A filter on Relay 1 (Filter C), which has been set up to select calls with a called DTE address field of 2345 *, matches this call (see Section 18.1.3).
3. This filter is associated with the relay client that Relay 1 is using to switch calls to PSDN1.
4. Relay 1 makes a call to DTE 23450002 (the DTE address originally specified by System 1), using the DTE class specified by this relay client. In this case, the DTE class contains just one DTE, 23450001.

The mechanism for the reverse situation (System 2 making a call to System 1) is similar. However, in this case, System 2 must specify a called DTE address of 23450001 in order for PSDN1 to correctly make the connection.

1. System 2 makes an X.25 call to Relay 1, using PSDN1. The destination DTE address specified in this call is 23450001. Because System 2 cannot specify the final DTE address (that of System 1), it must use a different call parameter to indicate that the call is to be relayed to System 1. For example, it could set the call user data to a value of %XFE.
2. A filter on Relay 1, which has been set up to select calls with a call data value of %XFE, matches this call (see Section 18.1.3).
3. This filter is associated with the relay client that Relay 1 is using to switch calls to System 1.

Figure 19–3 DECNIS Switching Calls Locally



4. Relay 1 makes a call to System 1 using the DTE class specified by this relay client; in this case, the DTE class contains just one DTE, 1234. As the X.25 network in this case is a point-to-point link, the DTE address of System 1 need not be specified.

19.3.2 Switching Calls Across a DECnet/OSI Network

For System 1 to call System 3 or System 4, the mechanism is similar to that described above. However, the relay client for System 3 or System 4 cannot specify a local DTE class, since there are no DTEs on Relay 1 that can communicate with System 3 or System 4.

In this case, the relay client specifies a remote DTE class: this is a DTE class on Relay 2 that can be used to reach System 3 or System 4.

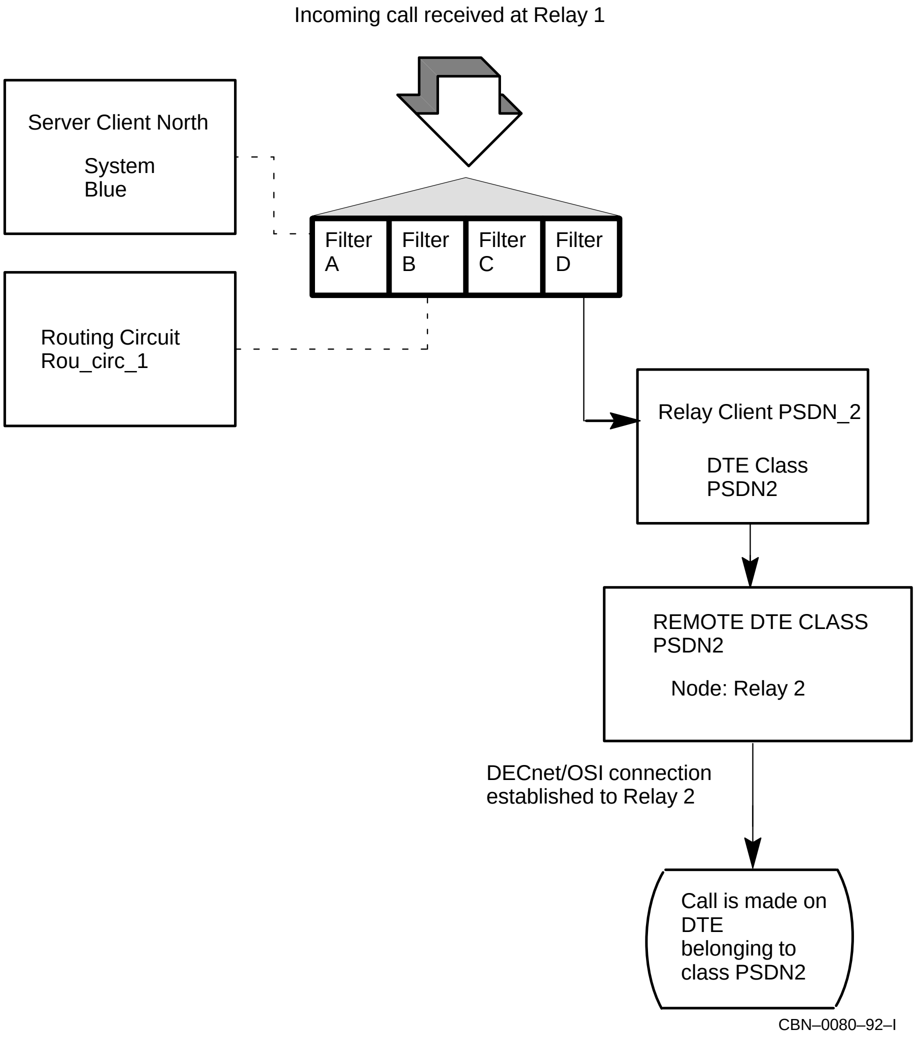
For example, Figure 19–4 shows the mechanism for System 1 to communicate with System 4:

1. System 1 makes an X.25 call to Relay 1, using its point-to-point link to Relay 1's DTE 1234. The destination DTE address specified in this call is 45670002.
2. A filter on Relay 1 (Filter D), which has been set up to select calls with a called DTE address field of 4567 *, matches this call (see Section 18.1.3).
3. This filter is associated with the relay client that Relay 1 is using to switch calls to PSDN2.
4. In this case, the DTE class specified by Relay Client PSDN_2 is a remote DTE class.
5. There must be a DTE CLASS entity of type Remote on Relay 1 called PSDN2. This must, in turn, specify a system on which a local DTE class of this name exists.
6. A DECnet/OSI connection is established between Relay 1 and the system named in remote DTE class PSDN2 (Relay 2).
7. Relay 2 then makes an X.25 call using the DTE class specified (PSDN2). The destination DTE address used for this call is 45670002, the original destination specified by System 1.

19.3.3 Using X.25 Reachable Addresses to Translate NSAP Addresses

If, in Figure 19–2, the X.25 application on System 1 that is making a call to System 2 specifies the NSAP address of System 2, rather than the DTE address, you must set up an X.25 reachable address on the DECNIS that maps this NSAP address on to the DTE address of System 2. Section 19.10 describes X.25 reachable addresses.

Figure 19–4 DECNIS Switching Calls Remotely



19.4 Managing Relay Clients

You must set up a relay client for every DTE or PSDN to which X.25 calls are to be switched.

19.4.1 Adding a Relay Client

1. Ensure that the X25 RELAY entity exists by entering the following command:

   ```
   NCL> CREATE X25 RELAY
   ```

2. Create the RELAY CLIENT entity for the relay client system:

   ```
   NCL> CREATE X25 RELAY CLIENT client-name
   ```

   where *client-name* is a name to identify the relay client. You should base this name on the node name or the PSDN name to which calls are to be switched.

3. Associate one or more filters with this relay client:

   ```
   NCL> SET X25 RELAY CLIENT client-name FILTERS {filter1,filter2,...}
   ```

   where *filter1* and *filter2* are filters that will be used to select calls for this relay client. Section 18.5 describes filters and how to manage them.

4. Specify the DTE class to be used to relay calls to this relay client:

   ```
   NCL> SET X25 RELAY CLIENT client-name DTE CLASS dte-class
   ```

   where *dte-class* is the name of a DTE class that exists on your system.

   If the relay client system is to be reached using a DTE on the DECNIS, the DTE class must be of type Local. Section 18.3 describes local DTE classes, and how to manage them.

   If the relay client system is to be reached by a DTE on another X.25 relay, in the same DECnet/OSI network as the DECNIS, then the DTE class must be of type Remote. Section 19.5 describes remote DTE classes, and how to manage them.

5. If required, associate a template with the relay client; this is used to specify any call parameters that are not specified in the original call request packet:

   ```
   NCL> SET X25 RELAY CLIENT client-name TEMPLATE template-name
   ```

   In the case of a remote relay, the template you specify must exist on the local system, that is, the same DECNIS on which the RELAY CLIENT entity exists. Section 9.14 describes templates and how to manage them.

6. Set up security for the relay client; see Section 20.6.
7. Create an X.25 reachable address for the relay client if required, to translate target NSAP addresses to target DTE addresses; Section 19.10 describes X.25 reachable addresses and how to manage them.
8. Enable the relay client:

   ```
   NCL> ENABLE X25 RELAY CLIENT client-name
   ```

19.4.2 Deleting a Relay Client

Delete a relay client if you want to prevent permanently any calls from being switched to the DTE or PSDN associated with that relay client. Use the DISABLE command if you want to prevent temporarily any calls from being switched by that relay client.

1. Clear any SVCs that have been established by the relay client:

   ```
   NCL> CLEAR X25 ACCESS PORT *, WITH CLIENT = client-name
   ```

   where *client-name* is the name of the relay client you want to delete.
2. Disable and delete the X25 RELAY CLIENT entity:

   ```
   NCL> DISABLE X25 RELAY CLIENT client-name
   NCL> DELETE X25 RELAY CLIENT client-name
   ```

3. Now delete the filter(s) that were associated with that relay client: see Section 18.5.3.

19.4.3 Changing the List of Filters Associated with a Relay Client

1. Disable the X25 RELAY CLIENT entity:

   ```
   NCL> DISABLE X25 RELAY CLIENT client-name
   ```

2. Add a filter to the relay client as follows:

   ```
   NCL> ADD X25 RELAY CLIENT client-name FILTERS {filter-name}
   ```

3. Remove a filter from the relay client as follows:

   ```
   NCL> REMOVE X25 RELAY CLIENT client-name FILTERS {filter-name}
   ```

   Note that removing a filter will have no effect on current calls that were established using that filter.
4. Enable the X25 RELAY CLIENT entity:

   ```
   NCL> ENABLE X25 RELAY CLIENT client-name
   ```

19.5 Managing Remote DTE Classes

19.5.1 Introduction

A DTE CLASS entity of type Remote is used to specify a DTE class on a remote system in the same DECnet/OSI network as the DECNIS. The name of the remote system on which the DTE class exists locally is associated with the remote DTE CLASS entity on the DECNIS.

19.5.2 Adding a Remote DTE Class

1. Ensure that the X25 ACCESS entity exists by entering the following command:

   ```
   NCL> CREATE X25 ACCESS
   ```

2. Create the DTE CLASS entity, specifying that it is of type Remote:

   ```
   NCL> CREATE X25 ACCESS DTE CLASS class-name TYPE REMOTE
   ```

3. Specify the system on which a local DTE called *class-name* exists:

   ```
   NCL> SET X25 ACCESS DTE CLASS class-name NODE node
   ```

   where *node* is the node name, as defined in the Known Towers database (see Section 1.11), of the X.25 system that has a local DTE class called *class-name*.

19.5.3 Deleting a Remote DTE Class

Delete a remote DTE class in exactly the same way as deleting a local DTE class: see Section 18.3.3.

19.5.4 Changing the System Associated with a Remote DTE Class

Specify the new system using the following command:

```
NCL> SET X25 ACCESS DTE CLASS class-name NODE node
```

where *node* is a node name defined in the Known Towers database; see Section 1.11.

19.6 Creating a Permanent Association Between Two DTEs

This section describes how to create a permanently switched connection between two X.25 systems. You do this by setting up a Permanent Virtual Circuit (PVC—see Section 18.6) on each of the X.25 connections, and setting up a relay PVC on the DECNIS.

Figure 19–5 and Figure 19–6 show two possible configurations. In Figure 19–5, PVCs are set up on DTE1 and DTE2. Data between System 1 and System 2 is permanently switched by setting up a relay PVC on the DECNIS. The relay PVC specifies the names of both PVCs.

In Figure 19–6, a PVC is set up on DTE1 on Relay 1. Another PVC is set up on DTE2 on Relay 2. Again, a relay PVC is set up on Relay 1, specifying the names of both PVCs. However, because one of the PVCs is on a remote system, the relay PVC must also specify the node name of the remote system. It does this by specifying a remote DTE class, which in turn specifies the node name of System 2.

Figure 19–5 Switching Between PVCs Locally

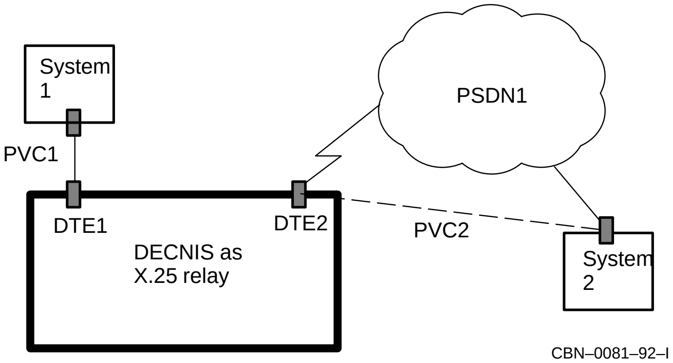
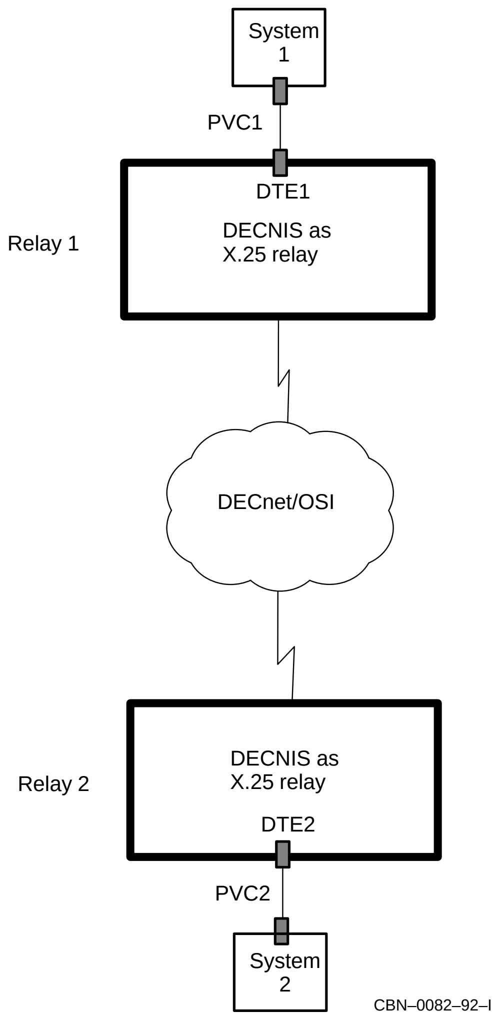


Figure 19–6 Switching Between PVCs Remotely



19.7 Managing Relay PVCs

19.7.1 Introduction

This section describes how to set up a permanently switched connection between two DTEs. You must set up a relay PVC for every pair of DTEs that are to be connected in this way. You must also set up a PVC for each X.25 connection; Section 18.6 describes PVCs and how to manage them.

19.7.2 Adding a Relay PVC

1. Ensure that the X25 RELAY entity exists by entering the following command:

   ```
   NCL> CREATE X25 RELAY
   ```

2. Create the RELAY PVC entity for the pair of DTEs:

   ```
   NCL> CREATE X25 RELAY PVC relay-pvc-name
   ```

   where *relay-pvc-name* is a name to identify the relay PVC.

3. Specify the name of the local PVC, that is, the one that exists on your DECNIS. If both PVCs exist on your DECNIS, as in Figure 19–5, enter either one here:

   ```
   NCL> SET X25 RELAY PVC relay-pvc-name LOCAL PVC local-pvc
   ```

4. Enter the name of the other PVC involved in the connection. In Figure 19–6, this would be PVC2.

   ```
   NCL> SET X25 RELAY PVC relay-pvc-name RELAYED PVC remote-pvc
   ```

5. If the PVC called *remote-pvc* does not exist on your DECNIS (as in Figure 19–6), you must associate a remote DTE class with this relay PVC:

   ```
   NCL> SET X25 RELAY PVC relay-pvc-name DTE CLASS class-name
   ```

   where *class-name* is the name of a remote DTE class that exists on your system. This remote DTE class must specify the node name of the system on which the remote relayed PVC exists. Section 19.5 describes remote DTE classes, and how to manage them.

6. Set up security for the relay PVC; see Section 20.6.

7. Enable the relay PVC:

   ```
   NCL> ENABLE X25 RELAY PVC client-name
   ```

Note that only one relay PVC should exist for each permanently switched connection. For example, in Figure 19–6, you should create the RELAY PVC entity only on Relay 1 or Relay 2, not on both.

19.7.3 Deleting a Relay PVC

Delete a relay PVC if you want to prevent permanently the PVCs from being used as a permanently switched connection between two DTEs. Use the DISABLE command if you want to prevent temporarily the PVCs from being used in this way.

Disable and delete the X25 RELAY PVC entity:

```
NCL> DISABLE X25 RELAY PVC relay-pvc-name
NCL> DELETE X25 RELAY PVC relay-pvc-name
```

19.8 LLC2 Communication and the CONS LAN/WAN Relay

19.8.1 Introduction

Figure 19–7 shows the DECNIS acting as a LAN/WAN relay. In this configuration, Systems 1, 2, and 3 are using LLC2 data links to make X.25 connections to the DECNIS. The LLC2 protocol is used, instead of LAPB, when X.25 communication is to take place over a LAN.

The DECNIS uses its relay function to switch the X.25 calls from the host systems on the LAN to the PSDN, and from the PSDN (Systems 4 and 5) to the LAN. In this way, there is a Connection-Oriented Network Service (CONS) between the LAN and the WAN (the PSDN in this case).

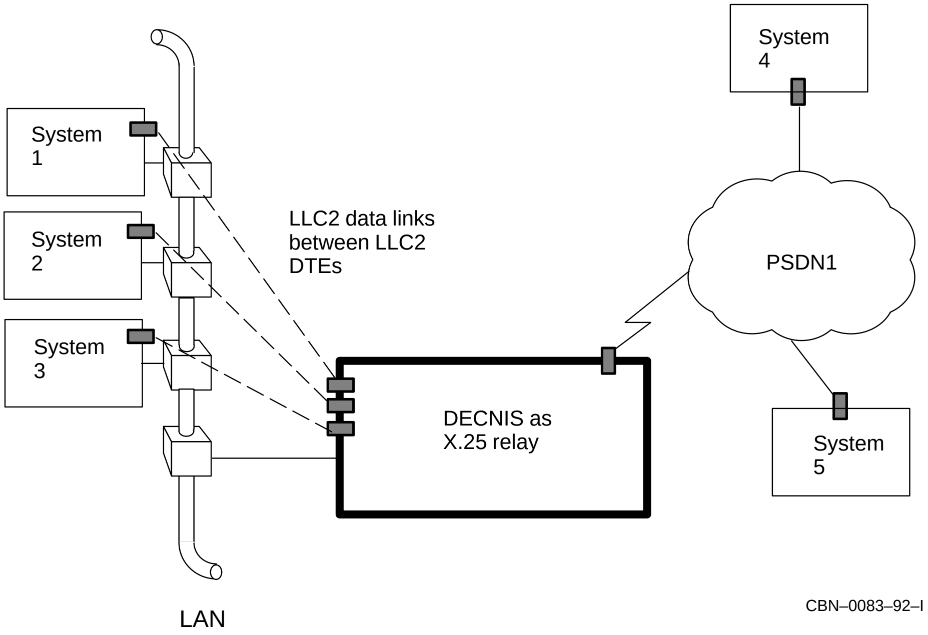
In addition, the DECNIS can also switch calls between the host systems on the LAN, for example, between System 1 and System 2.

19.8.2 Operating the DECNIS as a CONS LAN/WAN Relay

This section describes how to set up the CONS LAN/WAN relay function of the DECNIS.

1. Create an LLC2 DTE on the DECNIS for every system on the LAN with which you want to communicate; Section 19.9.3 describes how to set up LLC2 DTEs.

Figure 19–7 DECNIS Performing as a CONS LAN/WAN Relay



2. Create a relay client on the DECNIS for every host system or PSDN to which calls are to be switched. For example, in Figure 19–7, you would create four relay clients (one each for Systems 1 to 3 and one for PSDN1) if you wanted all systems to be able to communicate. Section 19.4.1 describes how to set up relay clients.
3. Create any X.25 reachable addresses required, to translate target NSAP addresses to target DTE addresses; Section 19.10 describes X.25 reachable addresses and how to manage them.

19.9 Managing LLC2 DTEs

19.9.1 Introduction

To communicate over a synchronous line using the X.25 recommendations, you set up a DTE that uses a LAPB data link (see Section 18.2.2). For X.25 communication over a LAN, you set up a DTE that uses an LLC2 data link.

In this way, X.25 at Level 3 can be used on a LAN. ISO 8881 defines how to send X.25 packets over an LLC2 data link.

You must set up an LLC2 DTE for every host system on the LAN with which you want to communicate. You can use the same LAN hardware port for all your LLC2 DTEs, or you can divide the LLC2 DTEs between two or more LAN hardware ports.

19.9.2 Setting Up a LAN Connection to Use for LLC2 DTEs

1. Create a LAN connection (for example, a CSMA/CD station) to use for the LLC2 DTEs, if a suitable one does not already exist. Section 8.7.2 explains how to create a CSMA/CD station.

   You can use the same LAN connection for all the LLC2 DTEs on the DECNIS (provided that all the remote systems with which you want to communicate are on the same LAN).

2. Create a SAP entity to represent this LAN connection as follows:

   ```
   NCL> CREATE LLC2 SAP sap-name
   ```

   where *sap-name* identifies the LAN connection to be used for the LLC2 DTEs associated with this LAN connection.

3. Associate this SAP with the LAN connection:

   ```
   NCL> SET LLC2 SAP sap-name LAN STATION station-name
   ```

   where *station-name* is the name of the LAN station.

   For a CSMA/CD connection, specify the LAN STATION as follows:

   ```
   NCL> SET LLC2 SAP sap-name LAN STATION -
   _NCL> CSMA-CD STATION csmacd-station-name
   ```

   where *csmacd-station-name* is the name of the CSMA/CD station (see Section 8.7.2).

   For an FDDI connection, specify the LAN STATION as follows:

   ```
   NCL> SET LLC2 SAP sap-name LAN STATION -
   _NCL> FDDI STATION fddi-station-name
   ```

   where *fddi-station-name* is the name of the FDDI station (see Section 8.8).

4. Specify an LSAP (Link Service Access Point) address for this SAP:

   ```
   NCL> SET LLC2 SAP sap-name LOCAL LSAP ADDRESS n
   ```

   where *n* is a hexadecimal number representing the LSAP address to be used by incoming calls on this LLC2 DTE. This value should be agreed with the remote LLC2 DTEs to which the LLC2 DTEs on this SAP will connect; ISO 8881 recommends that an LSAP address of 7E should be used for all LLC2 SAPs.

5. Now enable the SAP:

   ```
   NCL> ENABLE LLC2 SAP sap-name
   ```

19.9.3 Adding an LLC2 DTE

1. Create a SAP link for the LLC2 DTE:

   ```
   NCL> CREATE LLC2 SAP sap-name LINK link-name
   ```

   where *link-name* is used to identify the data link that this LLC2 DTE will use. You should use a name based on the name of the LLC2 DTE that will use the link.

2. Specify the LAN hardware address of the remote LLC2 DTE:

   ```
   NCL> SET LLC2 SAP sap-name LINK link-name -
   _NCL> REMOTE MAC ADDRESS nn-nn-nn-nn-nn-nn
   ```

   where *nn-nn-nn-nn-nn-nn* is six pairs of hexadecimal digits.

3. Specify the remote LSAP address:

   ```
   NCL> SET LLC2 SAP sap-name LINK link-name REMOTE LSAP ADDRESS n
   ```

   where *n* is a hexadecimal number representing the LSAP address to be used by outgoing calls on this LLC2 DTE. This value should be agreed with the remote LLC2 DTE to which this LLC2 DTE will connect; ISO 8881 recommends that an LSAP address of 7E should be used for all LLC2 SAPs.

4. Enable the SAP link:

   ```
   NCL> ENABLE LLC2 SAP sap-name LINK link-name
   ```

5. Now create the LLC2 DTE entity:

   ```
   NCL> CREATE X25 PROTOCOL DTE dte-name PROFILE "ISO8881"
   ```

   ISO8881 is the profile to be used by all LLC2 DTEs; Section 18.2.2 explains DTE profiles.

6. Associate this LLC2 DTE with the data link created above:

   ```
   NCL> SET X25 PROTOCOL DTE dte-name LINK SERVICE PROVIDER -
   _NCL> LLC2 SAP sap-name LINK link-name
   ```

7. Specify the DTE address of this DTE (as agreed with the remote DTE with which this DTE will communicate):

   ```
   NCL> SET X25 PROTOCOL DTE dte-name X25 ADDRESS dte-address
   ```

   where *dte-address* is up to 15 decimal digits.

8. Specify the DTE class with which calls received on this DTE will be associated.

   ```
   NCL> SET X25 PROTOCOL DTE DTE-name INBOUND DTE CLASS dte-class
   ```

   You should make *dte-class* the same as the DTE class to which this DTE belongs (see Section 18.3).

9. Allocate channels for incoming and outgoing calls, in agreement with the remote DTE. For example, assume that you have agreed with the remote DTE to use the following channels on this DTE:

   - 1 to 32 both-way
   - 49 to 64 outgoing only
   - 65 to 512 incoming only

   Specify those channels that are to be used for outgoing calls; include both-way channels:

   ```
   NCL> SET X25 PROTOCOL DTE dte-name OUTGOING LIST [[1..32],[49..64]]
   ```

   Now specify those channels that are to be used for incoming calls; again, include both-way channels:

   ```
   NCL> SET X25 PROTOCOL DTE dte-name INCOMING LIST [[1..32],[65..512]]
   ```

10. Add the DTE to a DTE class on your system: see Section 18.3. Note that, for LLC2 DTEs, you should normally have one DTE class per LLC2 DTE, since each LLC2 DTE is associated with a specific remote LLC2 DTE.

11. Enable the DTE:

    ```
    NCL> ENABLE X25 PROTOCOL DTE dte-name
    ```

19.9.4 Deleting an LLC2 DTE

Delete an LLC2 DTE if you want to prevent permanently the DECNIS from communicating with the associated remote LLC2 DTE. Use the DISABLE command if you want to prevent temporarily the DECNIS from communicating with the associated remote LLC2 DTE.

1. Disable and delete the DTE:

   ```
   NCL> DISABLE X25 PROTOCOL DTE dte-name
   NCL> DELETE X25 PROTOCOL DTE dte-name
   ```

2. Amend the list of members of the DTE class of which this DTE was a member: see Section 18.3.4.

3. Delete the DTE class, if it is now empty: see Section 18.3.3.

4. Disable and delete the LLC2 SAP link used by this DTE:

```
NCL> DISABLE LLC2 SAP LINK link-name
NCL> DELETE LLC2 SAP LINK link-name
```

19.9.5 Changing LLC2 DTE Characteristics

Change LLC2 characteristics in the same way as changing synchronous DTE characteristics: see Section 18.2.10.

19.10 Managing X.25 Reachable Addresses

19.10.1 Introduction

Create X.25 reachable addresses on the DECNIS to translate target NSAP addresses into target DTE addresses.

Consider the situation in Figure 19–7. Suppose an application on System 1 wants to make an X.25 call to System 4. If the DTE address of System 4 is specified in the call packet, then the DECNIS can simply relay the call to System 4.

However, the calling application on System 1 may, instead of specifying the DTE address of System 4, specify the NSAP address of System 4 in the call packet. In this case, the DECNIS must translate this NSAP address into the DTE address of System 4, so that it can forward the call.

The DECNIS uses an X.25 reachable address to translate a target NSAP address to a target DTE address. You must set up an X.25 reachable address on the DECNIS for a relay client if calls for that relay client specify only the client's NSAP address, rather than its DTE address.

In the case of a remote relay, the reachable address must exist on the local system, that is, the same DECNIS on which the RELAY CLIENT entity exists.

19.10.2 Adding an X.25 Reachable Address

1. Create the X.25 reachable address as follows:

```
NCL> CREATE X25 ACCESS REACHABLE ADDRESS address-name -
_NCL> ADDRESS PREFIX nsap-prefix
```

where:

address-name is used to identify the reachable address. You should use a name based on the name of the relay client to which it refers.

nsap-prefix is a full or partial NSAP address to be translated to a DTE address. If a partial NSAP address is specified, then all calls whose target NSAP address begins with the NSAP prefix specified here will be forwarded to the same DTE. If a full NSAP address is specified, then there will be a one-to-one mapping between target NSAP address and target DTE address.

2. Specify that the DTE address is to be entered manually in this reachable address, and enter the address:

   ```
   NCL> SET X25 ACCESS REACHABLE ADDRESS address-name -
   _NCL> MAPPING MANUAL, DESTINATION dte-address
   ```

   where *dte-address* is the target DTE address to which calls with the NSAP prefix specified above will be forwarded.

3. Ensure that you do not enter a DTE class for the reachable address, since any DTE class you specify would override the DTE class specified in the relay client.

4. Create a template for the relay client (see Section 9.15), and set the NSAP Mapping characteristic to True:

   ```
   NCL> SET X25 ACCESS TEMPLATE template-name NSAP MAPPING TRUE
   ```

   This ensures that calls made using this relay client will use the X.25 reachable address created.

5. Associate this template with the relay client as described in Section 19.4.1.

19.10.3 Deleting an X.25 Reachable Address

Delete an X.25 reachable address to prevent permanently relay clients on the DECNIS from receiving calls bearing the relevant target NSAP address. Note that you cannot disable an X.25 reachable address.

Delete the X.25 reachable address as follows:

```
NCL> DELETE X25 ACCESS REACHABLE ADDRESS address-name
```

20

X.25 Security

20.1 Introduction

This chapter describes how to set up security for the X.25 functions of the DECNIS. It shows how to prevent unauthorized remote DTEs from accessing any Client systems of the DECNIS, and how to prevent any Client systems making unauthorized calls to remote DTEs.

20.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

20.1.2 Using the DECNIS Text-based Configurator

You can use the DECNIS text-based configurator to set up X.25 security. The clearVISN DECNIS configurator does not support X.25.

All tasks performed using the DECNIS text-based configurator assume that you have started the configurator on a suitable host on the network, selected Modify from the Main Menu, and selected the DECNIS you wish to modify from the DECNIS Node screen.

20.2 How Incoming Security Works

This section describes how to prevent remote DTEs from making unauthorized calls to destinations on the DECNIS. These destinations are either server clients or X.25 routing circuits.

Generally, X.25 security uses the remote DTE address contained in the call packet to decide if a call will be set up or cleared.

Incoming security has two parts:

1. The call acquires rights identifiers (see Figure 20–1).

2. These rights identifiers are then checked against the ACL (Access Control List) associated with the filter that matches the call (see Figure 20–2). The ACL specifies the type of access that the call can have to the server client or X.25 routing circuit associated with the filter: Section 20.7 describes ACLs.

In this way, only the remote DTEs that you authorize may access a filter, and the server client or X.25 routing circuit with which it is associated.

20.3 Setting Up Incoming Security

20.3.1 Introduction

Before setting up incoming security, you must decide the following for each filter on the DECNIS:

- Those remote DTEs that are to be denied access using a particular DTE class.
- Those remote DTEs that may access the filter using a particular DTE class, but only if the remote system pays for the call.
- Those remote systems that may access the filter using a particular DTE class, regardless of who pays for the call.

You must go through this process for each DTE class on the DECNIS; for example, you can allow a particular remote DTE to access a particular filter only if the call is received on a particular DTE class.

Note that you set up security for BCUGs by setting up security for the remote member of the BCUG, as for any other remote DTE.

Figure 20–1 How an Incoming Call Acquires Rights Identifiers

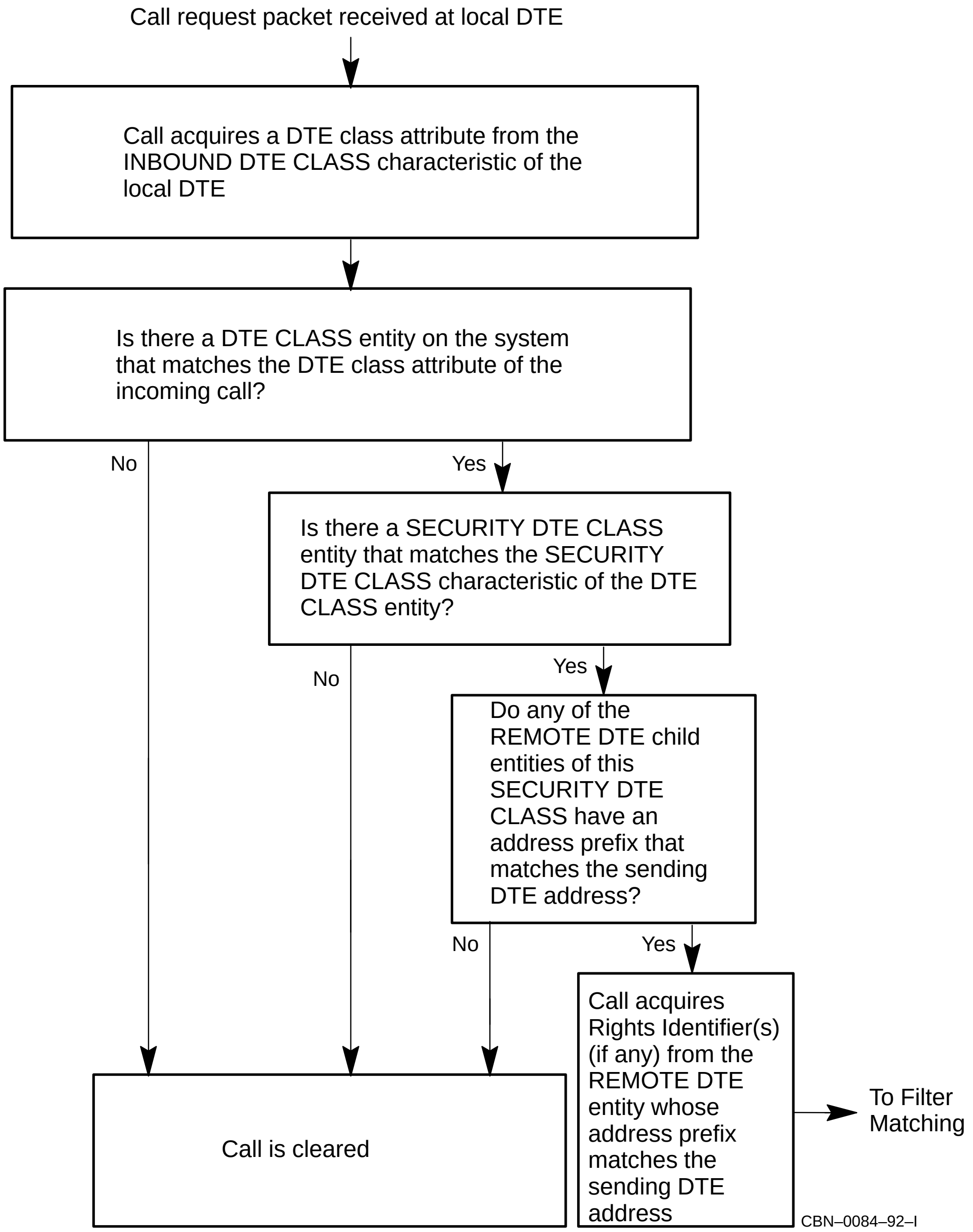
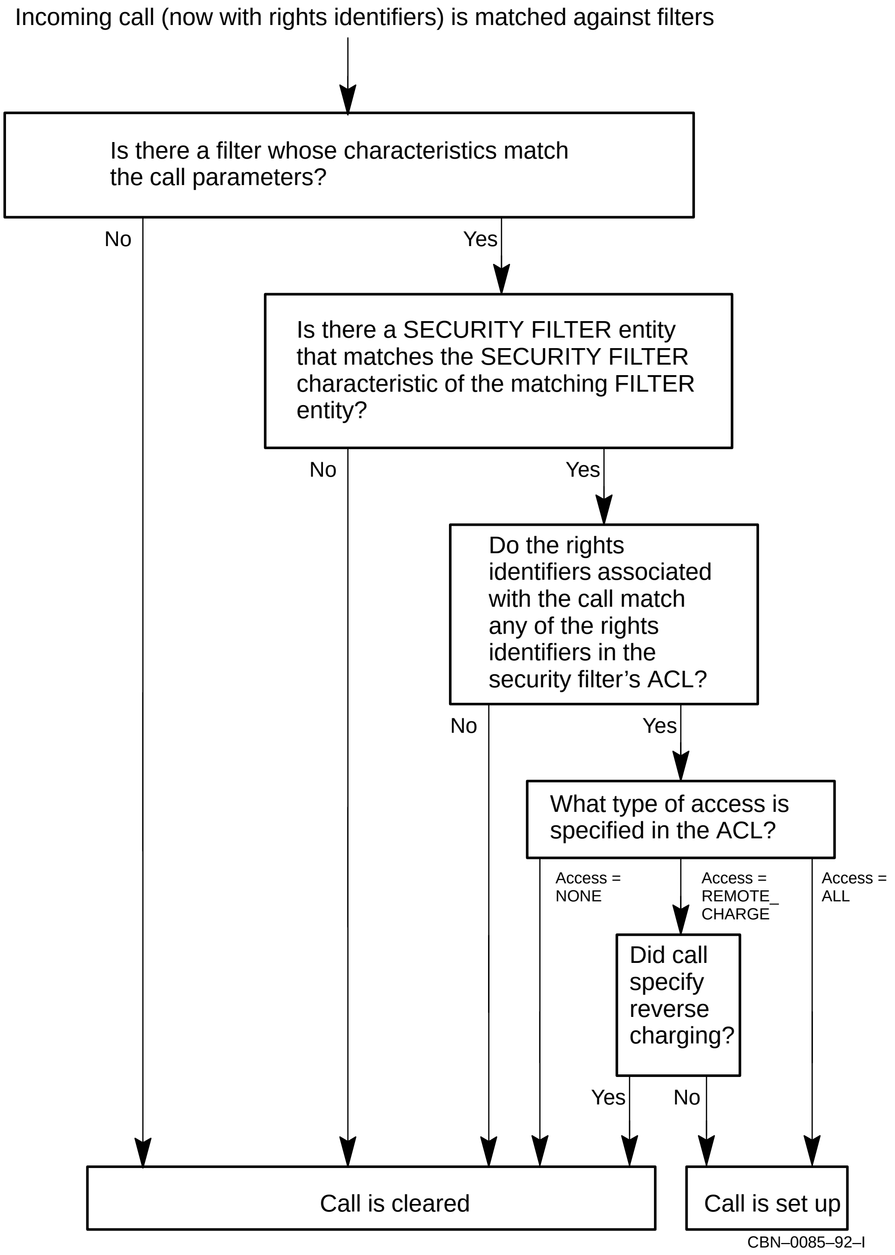

Call request packet received at local DTE
Call acquires a DTE class attribute from the INBOUND DTE CLASS characteristic of the local DTE
Is there a DTE CLASS entity on the system that matches the DTE class attribute of the incoming call?
No
Yes
Is there a SECURITY DTE CLASS entity that matches the SECURITY DTE CLASS characteristic of the DTE CLASS entity?
No
Yes
Do any of the REMOTE DTE child entities of this SECURITY DTE CLASS have an address prefix that matches the sending DTE address?
No
Yes
Call is cleared
Call acquires Rights Identifier(s) (if any) from the REMOTE DTE entity whose address prefix matches the sending DTE address
To Filter Matching
CBN–0084–92–I

Figure 20–2 How X.25 Security Decides Whether to Permit an Incoming Call


Incoming call (now with rights identifiers) is matched against filters
Is there a filter whose characteristics match the call parameters?
No
Yes
Is there a SECURITY FILTER entity that matches the SECURITY FILTER characteristic of the matching FILTER entity?
No
Yes
Do the rights identifiers associated with the call match any of the rights identifiers in the security filter's ACL?
No
Yes
What type of access is specified in the ACL?
Access = NONE
Access = REMOTE_ CHARGE
Access = ALL
Did call specify reverse charging?
Yes
No
Call is cleared
Call is set up
CBN–0085–92–I

20.3.2 Controlling Access to a Filter by a Remote DTE

This section describes how to grant free access and remote charge access to filters by remote DTEs. ‘Remote charge’ access means the remote DTE can access the filter only if the remote system pays for the call.

1. Create a SECURITY DTE CLASS entity:

   ```
   NCL> CREATE X25 ACCESS SECURITY DTE CLASS security-DTE-class
   ```

2. Specify the DTE classes for which this security DTE class will be used.

   ```
   NCL> SET X25 ACCESS DTE CLASS class-name -
   _NCL> SECURITY DTE CLASS security-DTE-class
   ```

   where *class-name* is the name of the DTE class on the DECNIS that will use this security DTE class. Issue this command for every DTE class for which this security DTE class will be used.

3. Create the REMOTE DTE child entities of this security DTE class, and specify a remote address prefix for each of them:

   ```
   NCL> CREATE X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE REMOTE ADDRESS PREFIX DTE-address
   ```

   where *DTE-address* is the address, or address prefix, of the remote DTE for which you want to set up security. If you specify an address prefix, the same security will apply to all remote DTEs whose DTE addresses begin with the prefix.

4. Decide on a rights identifier for the remote DTEs that are to have free access to Filter *filter*, and a rights identifier for those that are to have remote-charge access: for example, *filter*-FREE and *filter*-REMOTE.

5. Assign each rights identifier to the appropriate REMOTE DTE entities. For example:

   ```
   NCL> ADD X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE-1 -
   _NCL> RIGHTS IDENTIFIERS {filter-FREE}
   ```

   ```
   NCL> ADD X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE-2 -
   _NCL> RIGHTS IDENTIFIERS {filter-REMOTE}
   ```

6. Create a security filter for the filter on which you want to set up security:

   ```
   NCL> CREATE X25 ACCESS SECURITY FILTER security-filter
   ```

7. Associate this security filter with the filter on which you want to set up security:

   ```
   NCL> SET X25 ACCESS FILTER filter -
   _NCL> SECURITY FILTER security-filter
   ```

   If you want the access that remote DTEs have to a server client to be independent of which filter is used to match the call, you must associate this security filter with all the filters of a particular server client.

8. Set the ACL characteristic of the security filter to reflect the rights identifiers created above:

   ```
   NCL> SET X25 ACCESS SECURITY FILTER security-filter -
   _NCL> ACL ((IDENTIFIER = (filter-FREE), ACCESS = ALL), -
   _NCL> (IDENTIFIER = (filter-REMOTE), ACCESS = REMOTE))
   ```

20.3.3 Preventing Access to a Filter by a Remote DTE

This section explains how to deny access to a filter by a DTE that previously had access.

1. Identify the REMOTE DTE entity that grants the remote DTE its rights identifiers.

2. If the REMOTE DTE entity is used to grant rights identifiers only to the rogue DTE, remove the rights identifiers by setting them to a null set:

   ```
   NCL> SET X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE RIGHTS IDENTIFIERS {}
   ```

3. If the REMOTE DTE entity is used to grant rights identifiers to a group of DTEs (including the rogue DTE), create a new REMOTE DTE entity, specifying the full DTE address of the rogue DTE:

   ```
   NCL> CREATE X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE rogue-DTE -
   _NCL> REMOTE ADDRESS PREFIX rogue-DTE-address
   ```

   Do not set any rights identifiers for this new REMOTE DTE entity.

20.3.4 Creating a Closed, Semiclosed, or Open System

This section describes how to set up security on a filter so that:

- All remote DTEs, except those specified, are denied access (a closed system).
- All remote DTEs, except those specified, can access the filter provided the remote system pays for the call (a semiclosed system).

- All remote DTEs, except those specified, can access the filter, regardless of who pays for the call (an open system).

20.3.4.1 Procedure

1. Create a REMOTE DTE entity that will match calls from any unspecified DTEs:

   ```
   NCL> CREATE X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE REMOTE ADDRESS PREFIX *
   ```

2. Add a rights identifier that will be granted to all of these unspecified DTEs; for example, "default".

   ```
   NCL> SET X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE RIGHTS IDENTIFIERS {default}
   ```

3. Now set up security for the filter as in Section 20.3.2. However, in step 8, add an entry to the end of the ACL, as follows:

   - To set up a closed system, add the following ACE:

     ```
     (IDENTIFIER = (default), ACCESS = NONE)
     ```

   - To set up a semiclosed system, add the following ACE:

     ```
     (IDENTIFIER = (default), ACCESS = REMOTE_CHARGE)
     ```

   - To set up an open system, add the following ACE:

     ```
     (IDENTIFIER = (default), ACCESS = ALL)
     ```

   Because the ACEs are checked in the order they appear in the ACL, you must add these entries to the end of the ACL.

Note that, if you set up an open system, all remote DTEs can access this filter, and you may have to pay for the calls.

20.3.5 Applying the Same Incoming Security to All Filters on a System

Apply the same security to all the filters on a system by setting the SECURITY FILTER characteristic of each filter to the same value.

If all filters are protected by the same security filter, then each remote DTE has the same rights of access to every filter, and thus every server client and X.25 routing circuit, on your system.

If you have used the configurator to set up incoming security, all filters associated with a particular server client will be protected by a single security filter, based on the name of the server client. To use the configurator to set up identical security for each server client, supply information as follows:

	Screen	Action
1	Sections Menu	Select Incoming Security: X.25 Server Clients.
2	Incoming Security: X.25 Server Clients Options Menu	Select Copy Security Information From an X.25 Server Client.
3	Copy Incoming Security: X.25 Server Clients	Select the server client whose incoming security is to be used for all other server clients.
4		Select another server client, to which this security information will be copied.
5	Incoming Security: X.25 Server Clients Options Menu	Select Copy Security Information From an X.25 Server Client, and select another server client to copy security information to. Repeat this process until all server clients have the same incoming security information.

20.3.6 Applying the Same Incoming Security to All Local DTE Classes

If you set up several security DTE classes, and specify a different one for each DTE class, then the decision on whether X.25 Security allows a call to access a particular filter can depend on the local DTE on which it was received.

If you set up just one security DTE class and specify that all DTE classes are to use it, then the same incoming security will apply to calls irrespective of the DTE class on which they were received.

If you have used the configurator to set up incoming security, all the DTE classes will use a single security DTE class, called "Default".

20.4 How Outgoing Security Works

This section describes how to prevent Client systems from using the DECNIS as a Connector system to make unauthorized calls to remote DTEs. It also describes how to prevent calls being relayed to specified remote DTEs.

For outgoing security, each Client system that can use the DECNIS as a Connector system is granted rights identifiers. These rights identifiers are then checked against the ACL (see Section 20.7) of the remote DTE it wants to call. The ACL specifies the type of access the Client system can have to the remote DTE. You can specify:

- Those Client systems that can use the DECNIS as a Connector system.
- The remote DTEs to which they can make calls.
- Who must pay for the call.

Figure 20–3 How Outgoing Security Operates on the DECNIS

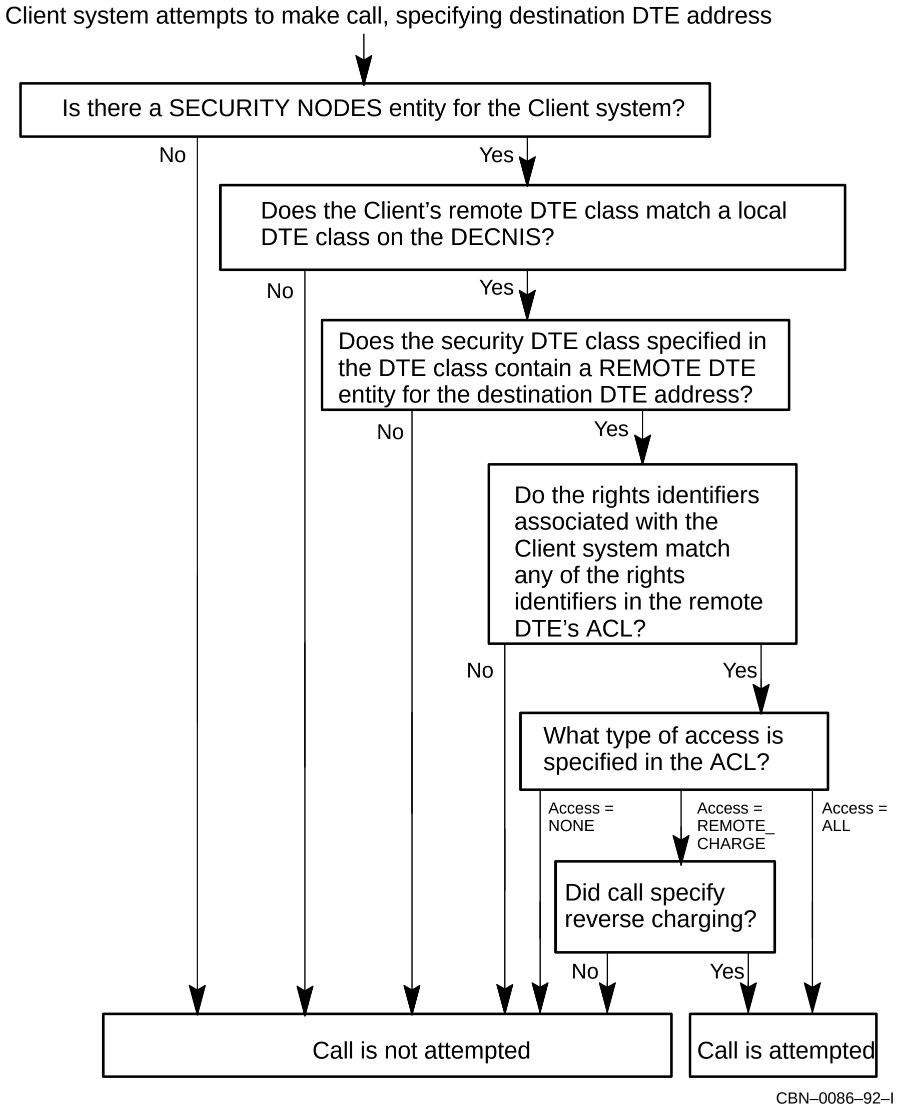
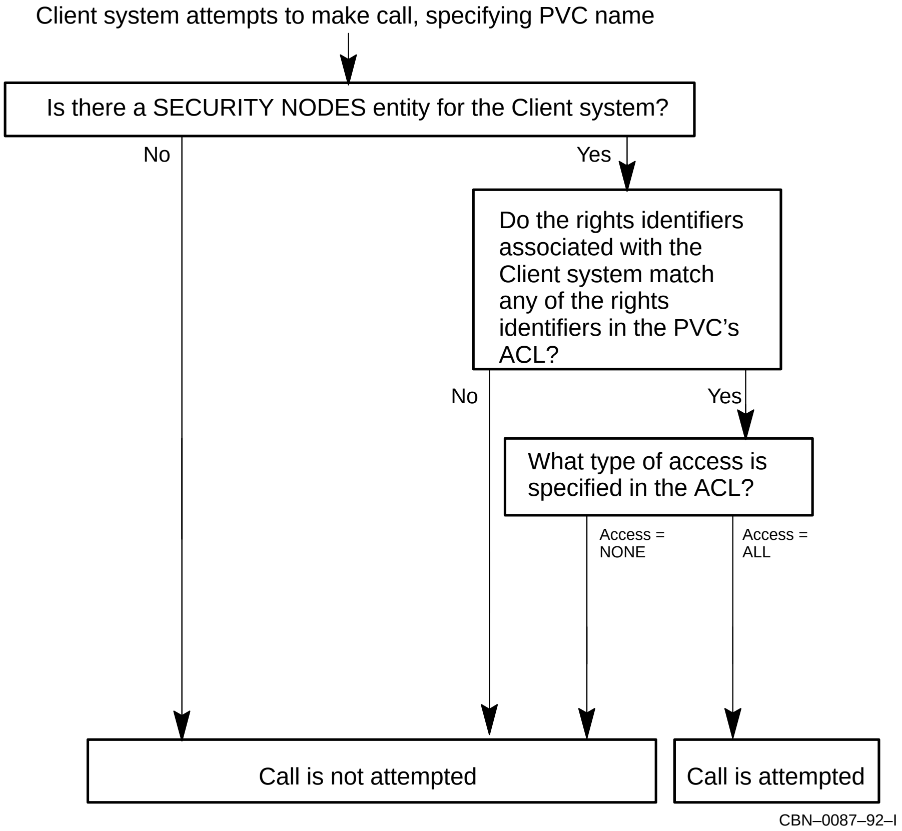

Client system attempts to make call, specifying destination DTE address
Is there a SECURITY NODES entity for the Client system?
No
Yes
Does the Client's remote DTE class match a local DTE class on the DECNIS?
No
Yes
Does the security DTE class specified in the DTE class contain a REMOTE DTE entity for the destination DTE address?
No
Yes
Do the rights identifiers associated with the Client system match any of the rights identifiers in the remote DTE's ACL?
No
Yes
What type of access is specified in the ACL?
Access = NONE
Access = REMOTE_ CHARGE
Access = ALL
Did call specify reverse charging?
No
Yes
Call is not attempted
Call is attempted
CBN–0086–92–I

Figure 20–3 illustrates outgoing security for X.25 gateway functions.

Figure 20–4 illustrates outgoing security on PVCs for X.25 gateway functions on the DECNIS.

Figure 20–4 How Outgoing Security for PVCs Operates on the DECNIS


Client system attempts to make call, specifying PVC name
Is there a SECURITY NODES entity for the Client system?
No
Yes
Do the rights identifiers associated with the Client system match any of the rights identifiers in the PVC's ACL?
No
Yes
What type of access is specified in the ACL?
Access = NONE
Access = ALL
Call is not attempted
Call is attempted
CBN–0087–92–I

20.5 Setting Up Outgoing Security

20.5.1 Introduction

Before setting up outgoing security, you must decide the following for each Client system:

- Those remote DTEs that cannot be called by the Client system using a particular DTE class.
- Those remote DTEs that can be called by the Client system using a particular DTE class, but only if the remote system pays for the call.
- Those remote DTEs that can be called by the Client using a particular DTE class, regardless of who pays for the call.
- Any PVCs on the DECNIS that can be accessed.
- Any PVCs on the DECNIS that cannot be accessed.

Note that you set up security for BCUGs by setting up security for the remote member of the BCUG, as for any other remote DTE.

20.5.2 Controlling Access to Remote DTEs by a Client System

This section describes how to grant free access and remote charge access to remote DTEs for a particular Client system. 'Remote charge' access means the Client system can call the remote DTE only if the remote system pays for the call.

1. Decide on the rights identifiers you want to give to the Client systems to control access. You should use a rights identifier based on the name(s) of the client system(s). For example, if you want to give *client-system1* and *client-system2* identical outgoing security, you could give them a rights identifier of *system1_system2_rights*.

2. Create a SECURITY NODES entity for the Client systems:

   ```
   NCL> CREATE X25 SERVER SECURITY NODES security-nodes-name
   NCL> SET X25 SERVER SECURITY NODES security-nodes-name -
   _NCL> NODES {client-system1, client-system2}
   ```

   where *client-system1* and *client-system2* are the client systems as defined in the Known Towers database (see Section 1.11).

   All the Client systems you enter in this SECURITY NODES entity will have identical outgoing security.

3. Give the Client system(s) their rights identifier:

   ```
   NCL> SET X25 SERVER SECURITY NODES security-nodes-name -
   _NCL> RIGHTS IDENTIFIERS {system1_system2_rights}
   ```

4. Create the security DTE class that will be used by the Client system(s):

   ```
   NCL> CREATE X25 ACCESS SECURITY DTE CLASS security-DTE-class
   ```

5. Now associate this security DTE class with the DTE class that the Client systems will use to access these remote DTEs:

   ```
   NCL> SET X25 ACCESS DTE CLASS class-name -
   _NCL> SECURITY DTE CLASS security-DTE-class
   ```

6. Create the REMOTE DTE entities for the remote DTEs that the Client systems will access:

   ```
   NCL> CREATE X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE1 -
   _NCL> REMOTE ADDRESS PREFIX address1

   NCL> CREATE X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE2 -
   _NCL> REMOTE ADDRESS PREFIX address2

   NCL> CREATE X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE3 -
   _NCL> REMOTE ADDRESS PREFIX address3
   ```

   where *address1*, *address2*, and *address3* are the full DTE addresses, or address prefixes, for the remote DTEs. If you specify a remote address prefix, then the same security will apply to all remote DTEs whose DTE addresses begin with the prefix.

7. Now construct ACLs using the rights identifier created above, and assign these to the appropriate REMOTE DTE entities.

   For example, if you want the Client systems in this SECURITY NODES entity to have free access to *remote-DTE1* and *remote-DTE2*, but remote charge access to *remote-DTE3*:

   ```
   NCL> SET X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE1 -
   _NCL> ACL ((IDENTIFIER = (system1_system2_rights), ACCESS = ALL))

   NCL>  SET X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE2 -
   _NCL> ACL ((IDENTIFIER = (system1_system2_rights), ACCESS = ALL))

   NCL>  SET X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE3 -
   _NCL> ACL ((IDENTIFIER = (system1_system2_rights), ACCESS = REMOTE))
   ```

20.5.3 Controlling Access to PVCs by a Client System

This section describes how to grant access to PVCs for a particular Client system.

1. Set up the SECURITY NODES entity, by following steps 1 to 3 of Section 20.5.2.
2. Set the ACL characteristic of the PVC entity. For example, if you want *client-system1* and *client-system2* to be able to access *PVC1*:

   ```
   NCL> SET X25 PROTOCOL DTE DTE-name PVC PVC1 -
   _NCL> ACL ((IDENTIFIER = (system1_system2_rights), ACCESS = ALL))
   ```

   Note that you can only permit or deny access to a PVC: remote-charge access does not exist for PVCs.

20.5.4 Preventing Access to a Remote DTE by a Client System

This section explains how to deny access to a remote DTE by a Client system that previously had access.

1. Identify the SECURITY NODES entity that is being used to grant rights identifiers to the Client system.
2. Add a rights identifier to this SECURITY NODES entity. For example, if you want to deny access to remote DTE *remote-DTE1* by this Client system, add the following rights identifier:

   ```
   NCL> ADD SECURITY NODES security-nodes-name -
   _NCL> RIGHTS IDENTIFIERS {no_access_remote-DTE1}
   ```

   (Note that this will prevent all the Client systems in this SECURITY NODES entity from accessing this DTE.)
3. Identify the REMOTE DTE entity that is used to control access to the remote DTE, for example, *remote-DTE1*.
4. Identify the ACL characteristic:

   ```
   NCL> SHOW X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE1 ACL
   ```

5. Amend the ACL by adding the following ACE to the top of the ACL:

   (Identifier = (no_access_*remote-DTE1*), Access = NONE)
6. Set this new ACL characteristic:

   ```
   NCL> SET X25 ACCESS SECURITY DTE CLASS security-DTE-class -
   _NCL> REMOTE DTE remote-DTE1 ACL ACL
   ```

20.5.5 Preventing a Client System from Using the DECNIS as a Connector System

This section assumes that a Client system has previously been authorized to make calls through the DECNIS. It explains how to prevent such a Client system from using the DECNIS as a Connector system to make outgoing calls.

Note that a Client system may be prevented from making calls, but can still use the DECNIS to receive calls. To prevent a Client system from receiving calls, delete the filter(s) associated with it on the DECNIS.

1. Identify the SECURITY NODES entity that grants rights identifers to the Client system.
2. If the only system specified in the NODES characteristic of the SECURITY NODES entity is the Client system, then delete the rights identifiers specified for the SECURITY NODES entity:

   ```
   NCL> SET X25 SERVER SECURITY NODES security-nodes-name -
   _NCL> RIGHTS IDENTIFIERS {}
   ```

3. If there are several systems specified in the NODES characteristic of the SECURITY NODES entity, remove the Client system from the set of nodes specified in the SECURITY NODES entity:

   ```
   NCL> REMOVE X25 SERVER SECURITY NODES security-nodes-name -
   _NCL> NODES {client-system}
   ```

   where *client-system* is the Client system as defined in the Known Towers database (see Section 1.11).

   If there is a catchall SECURITY NODES entity that is used to grant rights identifiers to all unspecified remote DTEs, you must create a new SECURITY NODES entity specifically for this Client system. Do not set any rights identifiers for this SECURITY NODES entity.

   ```
   NCL> CREATE X25 SERVER SECURITY NODES security-nodes-name
   ```

   ```
   NCL> SET X25 SERVER SECURITY NODES security-nodes-name -
   _NCL> NODES {client-system}
   ```

20.5.6 Creating a Closed, Semiclosed, or Open System

This section describes how to set up security so that:

- All Client systems, except those specified, are prevented from using the DECNIS as a Connector system (a closed system).
- All Client systems, except those specified, can use the DECNIS as a Connector system, provided the remote system pays for the call (a semiclosed system).

- All Client systems, except those specified, can use the DECNIS as a Connector system, regardless of who pays for the call (an open system).

20.5.6.1 Procedure

1. Create a default SECURITY NODES entity (called, for example, 'default_access'). This will be used by all Client systems that are not included in the NODES characteristic of any other SECURITY NODES entity:

   ```
   NCL> CREATE X25 SERVER SECURITY NODES default_access
   NCL> SET X25 SERVER SECURITY NODES default_access NODES {namespace:*...}
   ```

   where *namespace* is the name of the namespace of the nodes whose access to the DECNIS is to be controlled. Use the following form of command if there is more than one namespace:

   ```
   NCL> SET X25 SERVER SECURITY NODES default_access -
   _NCL> NODES {namespace1:*..., namespace2:*...}
   ```

2. Grant the Client systems using this SECURITY NODES entity a default rights identifier, 'default_rights':

   ```
   NCL> SET X25 SERVER SECURITY NODES default_access -
   _NCL> RIGHTS IDENTIFIERS {default_rights}
   ```

All Client systems can now use the DECNIS as a Connector system. You can specify the type of access that Client systems bearing the rights identifier 'default_rights' can have to remote DTEs and PVCs by amending the ACL characteristic of the appropriate REMOTE DTE and PVC entities.

20.5.7 Allowing Client Systems Outgoing Access to All Remote DTEs

This section describes how to allow a Client system to use the DECNIS to call any remote DTE.

1. Identify the SECURITY NODES entities for the Client systems concerned.

2. Add an identifier to these SECURITY NODES entities, for example:

   ```
   NCL> ADD X25 SERVER SECURITY NODES security-nodes-name -
   _NCL> RIGHTS IDENTIFIERS {can_access}
   ```

3. Create a REMOTE DTE entity, associated with the DTE class through which these Client systems will access the remote DTEs. For example, if the DTE class to be used is DTE-class1, and DTE-class1 specifies a security DTE class called DTE-security1, create a REMOTE DTE entity called 'all_access', and specify the remote address prefix as a wildcard:

   ```
   NCL> CREATE X25 ACCESS SECURITY DTE CLASS DTE-security1 -
   _NCL>  REMOTE DTE all_access REMOTE ADDRESS PREFIX *
   ```

4. Now set the ACL for this REMOTE DTE entity to allow access for Client systems bearing the rights identifier ‘can_access’:

```
NCL> CREATE X25 ACCESS SECURITY DTE CLASS DTE-security1 -
_NCL> REMOTE DTE all_access -
_NCL> ACL ((IDENTIFIER = (can_access), ACCESS = ALL))
```

5. To allow access for these Client systems only if the remote systems pay, set the access action of the ACL to REMOTE_CHARGE:

```
NCL> CREATE X25 ACCESS SECURITY DTE CLASS DTE-security1 -
_NCL> REMOTE DTE all_access -
_NCL> ACL ((IDENTIFIER = (can_access), ACCESS = REMOTE_CHARGE))
```

20.5.8 Applying the Same Outgoing Security to All Local DTE Classes

If you set up several security DTE classes, and specify different ones for each DTE class, then the decision on whether a Client system can call a particular remote DTE can depend on the DTE class that the Client system requested to use.

If you set up just one security DTE class and specify that all DTE classes are to use it, then the same outgoing security will apply to calls irrespective of the DTE class that a Client system requests to use.

If you have used the DECNIS text-based configurator to set up outgoing security, all the DTE classes will use the same security DTE class, called "Default".

20.6 X.25 Relay Security

20.6.1 How X.25 Relay Security Works

Incoming security for X.25 relay functions works as described in Section 20.2. You must set up X25 ACCESS SECURITY DTE CLASS REMOTE DTE entities for the remote DTEs that will make calls to the X.25 relay. In addition, you must set up X25 ACCESS SECURITY FILTER entities for the filters associated with the relay clients. Section 20.3 describes how to do this.

Outgoing security for X.25 relay functions is almost identical to that described in Section 20.4. Set up X25 ACCESS SECURITY DTE CLASS REMOTE DTE entities, and associated ACLs, as described in Section 20.5. However, the rights identifiers for an outgoing call are granted in three possible ways:

- If the outgoing X.25 call is to be made by the DECNIS on which the RELAY CLIENT entity exists, the rights identifiers are granted by the RELAY CLIENT entity.

- If the relay uses PVCs, and the RELAYED PVC characteristic of the RELAY PVC entity indicates a local PVC (that is, a PVC that exists on the same DECNIS as the RELAY PVC entity), the rights identifiers are granted by the RELAY PVC entity.
- If the DECNIS is relaying calls across a DECnet/OSI network to a remote relay system, the rights identifiers are granted by an X25 SERVER SECURITY NODES entity on the remote relay, and X.25 security on the remote relay is responsible for allowing or disallowing the call (see Section 20.5.2).

Figure 20–5 illustrates outgoing security for X.25 relay functions on the DECNIS.

For example, consider Figure 20–6. In this case, System 1 can make calls to System 3, but System 2 cannot. Set up security on the DECNIS as follows.

1. Set up incoming security to allow System 1, but not System 2, to access the filter(s) associated with the RELAY CLIENT entity corresponding to System 3 or PSDN1; see Section 20.3.2.
2. Grant the RELAY CLIENT entity corresponding to System 3 or PSDN1 a rights identifier; for example:

   ```
   NCL> SET X25 RELAY CLIENT psdn_1 RIGHTS IDENTIFIERS {system1_ok}
   ```

3. Now ensure that the ACL associated with the REMOTE DTE entity for System 3 specifies that access is granted to calls with the rights identifier SYSTEM1_OK. For example:

   ```
   NCL> SET X25 ACCESS SECURITY DTE CLASS security-class -
   _NCL> REMOTE DTE system_3 -
   _NCL> ACL ((IDENTIFIER = (system1_ok)), ACCESS = ALL)
   ```

Figure 20–5 How Outgoing Security for X.25 Relay Functions Operates on the DECNIS

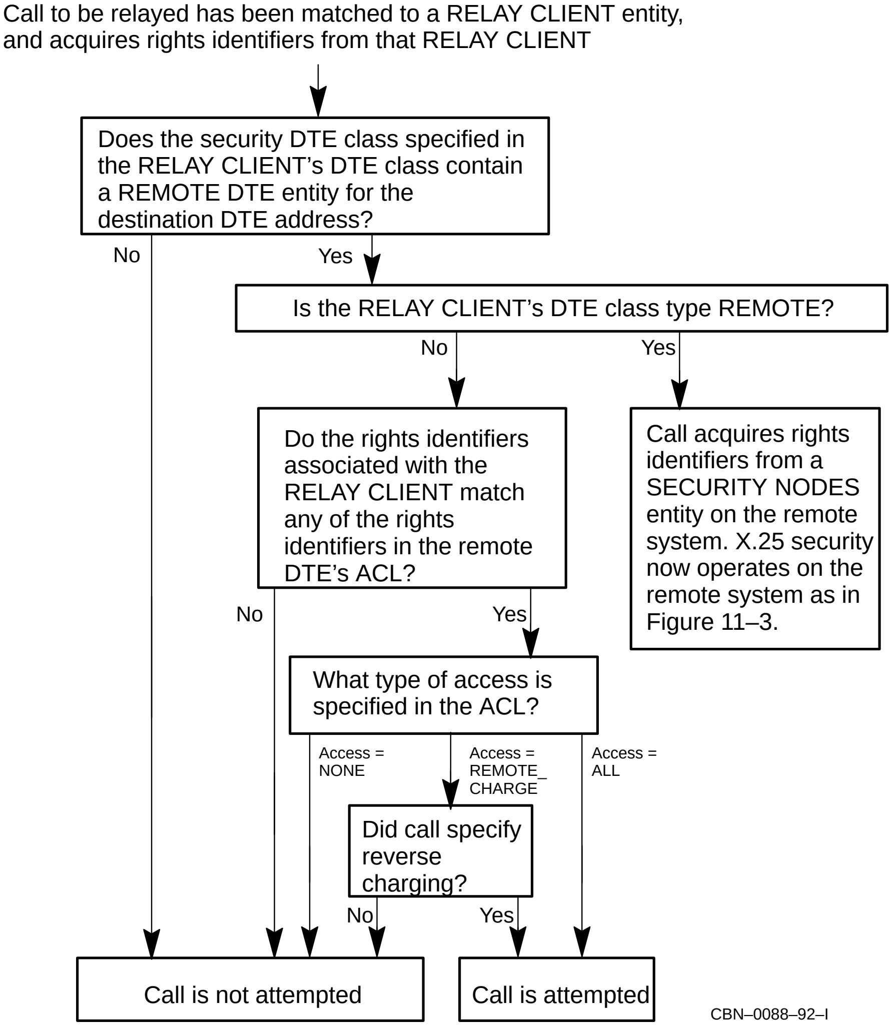
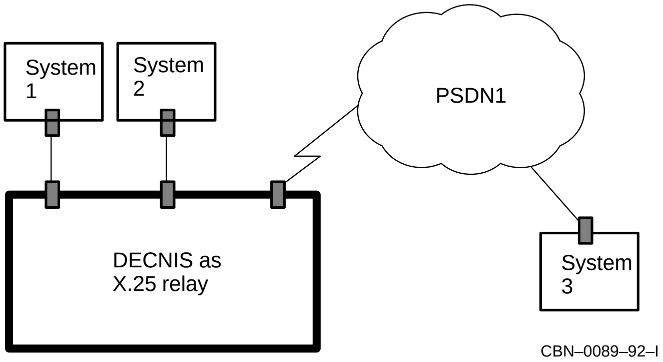

Call to be relayed has been matched to a RELAY CLIENT entity, and acquires rights identifiers from that RELAY CLIENT
Does the security DTE class specified in the RELAY CLIENT's DTE class contain a REMOTE DTE entity for the destination DTE address?
No
Yes
Is the RELAY CLIENT's DTE class type REMOTE?
No
Yes
Do the rights identifiers associated with the RELAY CLIENT match any of the rights identifiers in the remote DTE's ACL?
Call acquires rights identifiers from a SECURITY NODES entity on the remote system. X.25 security now operates on the remote system as in Figure 11–3.
No
Yes
What type of access is specified in the ACL?
Access = NONE
Access = REMOTE_ CHARGE
Access = ALL
Did call specify reverse charging?
No
Yes
Call is not attempted
Call is attempted
CBN–0088–92–I

Figure 20–6 X.25 Security for Relay Functions



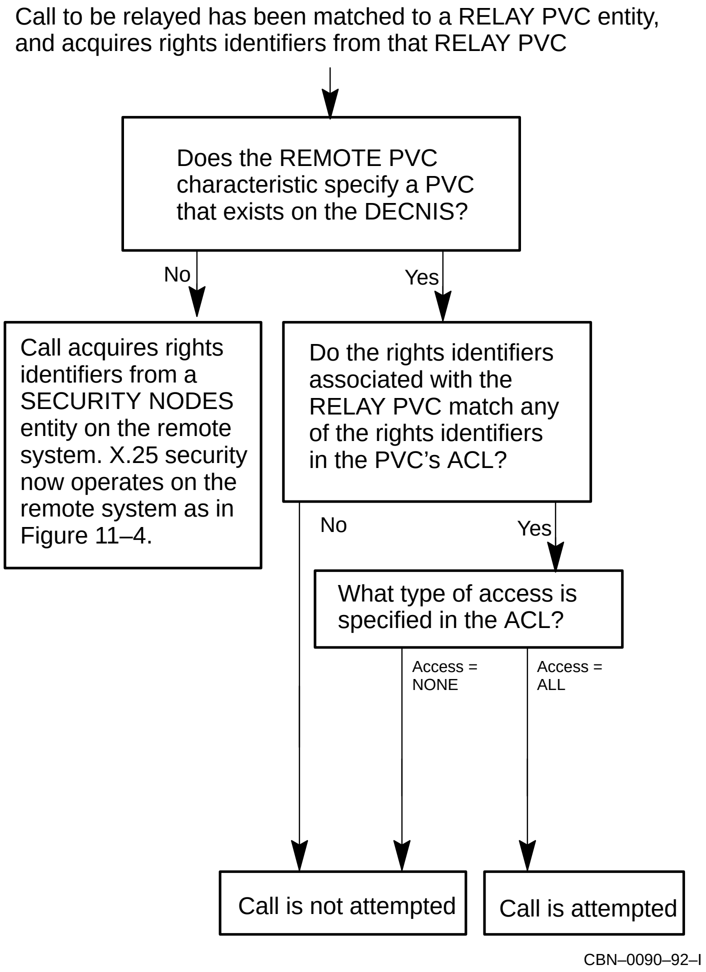
20.6.2 X.25 Relay Security and PVCs

Figure 20–7 illustrates outgoing security on PVCs for X.25 relay functions on the DECNIS.

In this case you assign rights identifiers to the relevant X25 RELAY PVC entity, as follows:

```
NCL> SET X25 RELAY PVC sys_3 RIGHTS IDENTIFIERS {system1_ok}
```

Figure 20–7 How Outgoing Security for X.25 Relay Functions on PVCs Operates on the DECNIS


Call to be relayed has been matched to a RELAY PVC entity, and acquires rights identifiers from that RELAY PVC
Does the REMOTE PVC characteristic specify a PVC that exists on the DECNIS?
No
Yes
Call acquires rights identifiers from a SECURITY NODES entity on the remote system. X.25 security now operates on the remote system as in Figure 11–4.
Do the rights identifiers associated with the RELAY PVC match any of the rights identifiers in the PVC's ACL?
No
Yes
What type of access is specified in the ACL?
Access = NONE
Access = ALL
Call is not attempted
Call is attempted
CBN–0090–92–I

20.7 ACLs

20.7.1 ACL Structure

An ACL (Access Control List) is a series of ACEs (Access Control Entries).

Each ACE is of the following form:

```
(Identifier = (identifier-list), Access = access)
```

identifier-list is a list of rights identifiers that you create and assign to a SECURITY NODES or REMOTE DTE entity.

access is one of:

- NONE

  Access is not permitted.

- REMOTE_CHARGE

  Access is only permitted if the remote system pays for the call.

- ALL

  Access is permitted, regardless of whether the local or remote system pays for the call.

20.7.2 Example ACL

A REMOTE DTE entity has the following ACL:

```
((Identifier = (north), Access = ALL),
 (Identifier = (south), Access = REMOTE_CHARGE),
 (Identifier = (*), Access = NONE))
```

Any Client system with the rights identifier ‘north’ can make calls to the remote DTE. It does not have to specify reverse charging.

Any Client system with the rights identifier ‘south’ can make calls to the remote DTE, provided it specifies reverse charging.

Any Client system that does not have the rights identifier ‘north’ or ‘south’ is prevented from making any calls to the remote DTE.

20.7.3 Order of Checking ACEs

The ACL is checked by X.25 security from top to bottom. As soon as a match is found, the checking process stops. Therefore, an ACE that has the wildcard as the identifier should always be at the bottom of an ACL.

For example, change the example in Section 20.7.2 so that the wildcard ACE is not at the bottom:

```
((Identifier = (north), Access = ALL),
 (Identifier = (*), Access = NONE),
 (Identifier = (south), Access = REMOTE_CHARGE))
```

This would mean that no process would ever be granted REMOTE_CHARGE access, since all rights identifiers would match the wildcard, and the checking process would never reach the final ACE.

Part IV

Managing Bridging

This part contains information on managing bridging on the DECNIS.

It contains the following chapters:

- Chapter 21 introduces the principles of bridging on the DECNIS, and describes how to set up bridge ports.
- Chapter 22 describes how to configure a bridge port to filter certain protocols and addresses.

21

Setting Up the DECNIS as a Bridge

21.1 Introduction

This chapter describes the principles of bridging on the DECNIS, and describes how to set up the DECNIS to perform local and remote bridging.

21.1.1 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

21.1.1.1 Managing Bridging from OpenVMS VAX Systems

On OpenVMS VAX hosts, you cannot use the standard version of NCL to manage the bridging functions of the DECNIS. Instead you must use a special bridge management version of NCL supplied with the DECNIS.

To start bridge management NCL on OpenVMS VAX hosts, enter the following command:

```
$ RUN SYS$SYSTEM:NIS$BRIDGE_MGMT.EXE
```

You can then enter bridge management NCL commands.

Access Control

When specifying NCL access control information to NIS$BRIDGE_MGMT.EXE, use one of the following syntaxes.

- Specify the user name and password as described in Section 2.13.4.3.
- Specify default as access as described in Section 2.13.4.4.

Note the following:

- You use this version of NCL to manage the BRIDGE module only on OpenVMS VAX hosts. On all other supported hosts, including OpenVMS Alpha hosts, you use the standard version of NCL to manage the BRIDGE module.
- On OpenVMS VAX systems, you run the standard version of NCL to manage all other modules.

21.2 Bridging

21.2.1 Definition

Bridging means linking two or more Local Area Networks (LANs) at the data link level to form an **extended LAN**.

A **bridge** is a server connected to two or more LANs that forwards frames according to source address, destination address, and protocol type.

21.2.2 Example

Figure 21–1 shows the DECNIS linking several LANs to form an extended LAN. The DECNIS is performing local and remote bridging on the extended LAN.

21.2.3 Local Bridging

Bridging is **local** when the DECNIS is directly connected to the LANs being bridged.

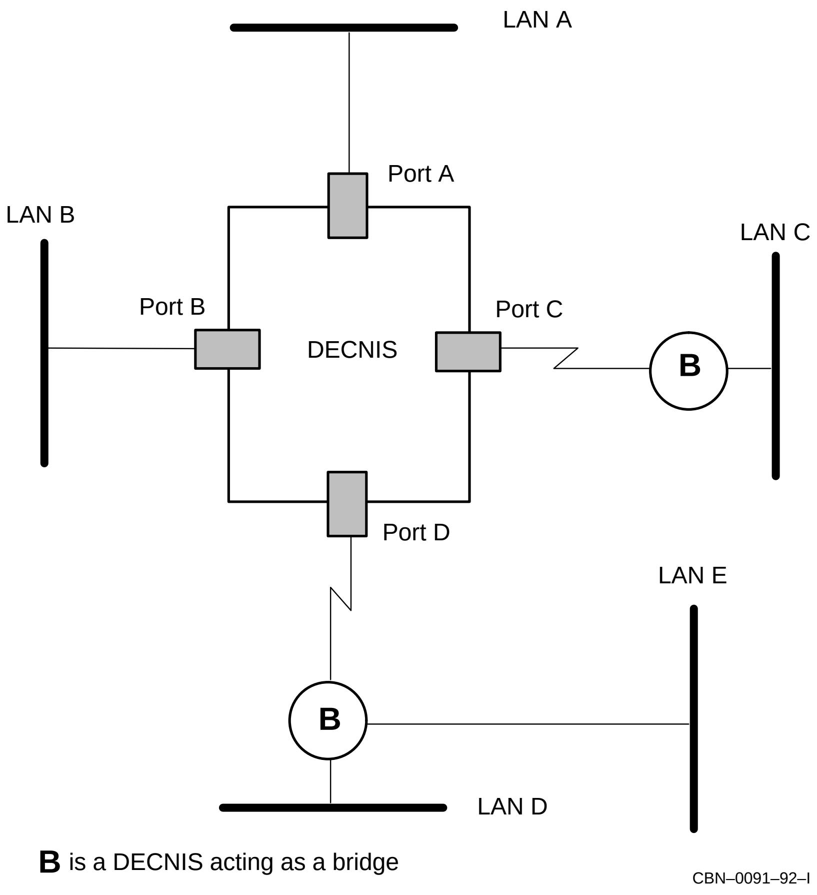
The bridge listens for frames on LAN A using its Bridge Port A. When it receives a frame on LAN A whose destination address is a system on LAN B, it forwards the frame on to LAN B using Bridge Port B. This illustrates local bridging.

21.2.4 Remote Bridging

Bridging is **remote** when the DECNIS is connected to the LANs being bridged over a synchronous data link.

When the bridge receives a frame whose destination is on LAN C, it forwards the frame on to LAN C using its remote Bridge Port C. This illustrates remote bridging.

Figure 21–1 DECNIS Operating as a Bridge



When it receives a frame whose destination is on LAN D or E, it forwards the frame on to the relevant LAN using its remote Bridge Port D.

21.2.4.1 Connecting to Vitalink TransLAN Bridges: VCP Bridging

When the remote bridge to which the DECNIS connects is a Vitalink TransLAN bridge, the data link used for the connection must be VCP. Section 8.9 describes how to set up a VCP data link.

21.2.5 Extended LANs: Definition

LANs A, B, C, D, and E in Figure 21–1 form an extended LAN. An extended LAN is a collection of LANs (sometimes known as LAN segments) interconnected by bridges.

Generally, a bridge port does not forward frames destined for the local LAN segment to which it is connected. This keeps the traffic on the extended LAN to a minimum.

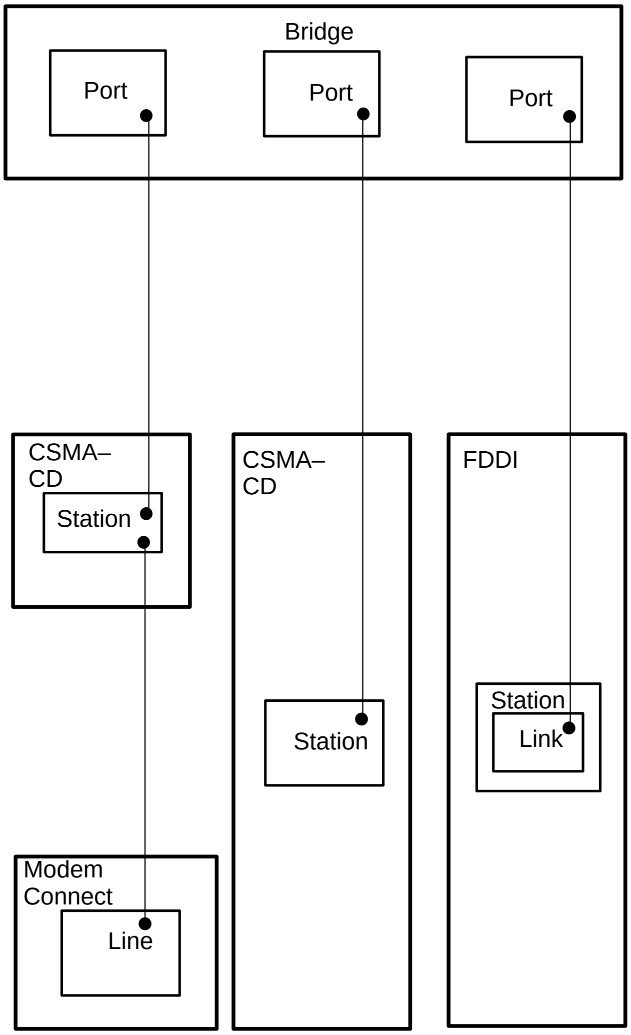
21.2.6 Further Reading

The *Bridge and Extended LAN Reference* manual provides further details of bridge operation and the spanning tree algorithm, as well as configuration and problem solving guidelines.

21.3 Entities Associated with a Bridge Port

Figure 21–2, Figure 21–3, and Figure 21–4 show the entities required by each type of bridge port.

Figure 21–2 Entities Required by Local and Remote VCP Bridge Ports

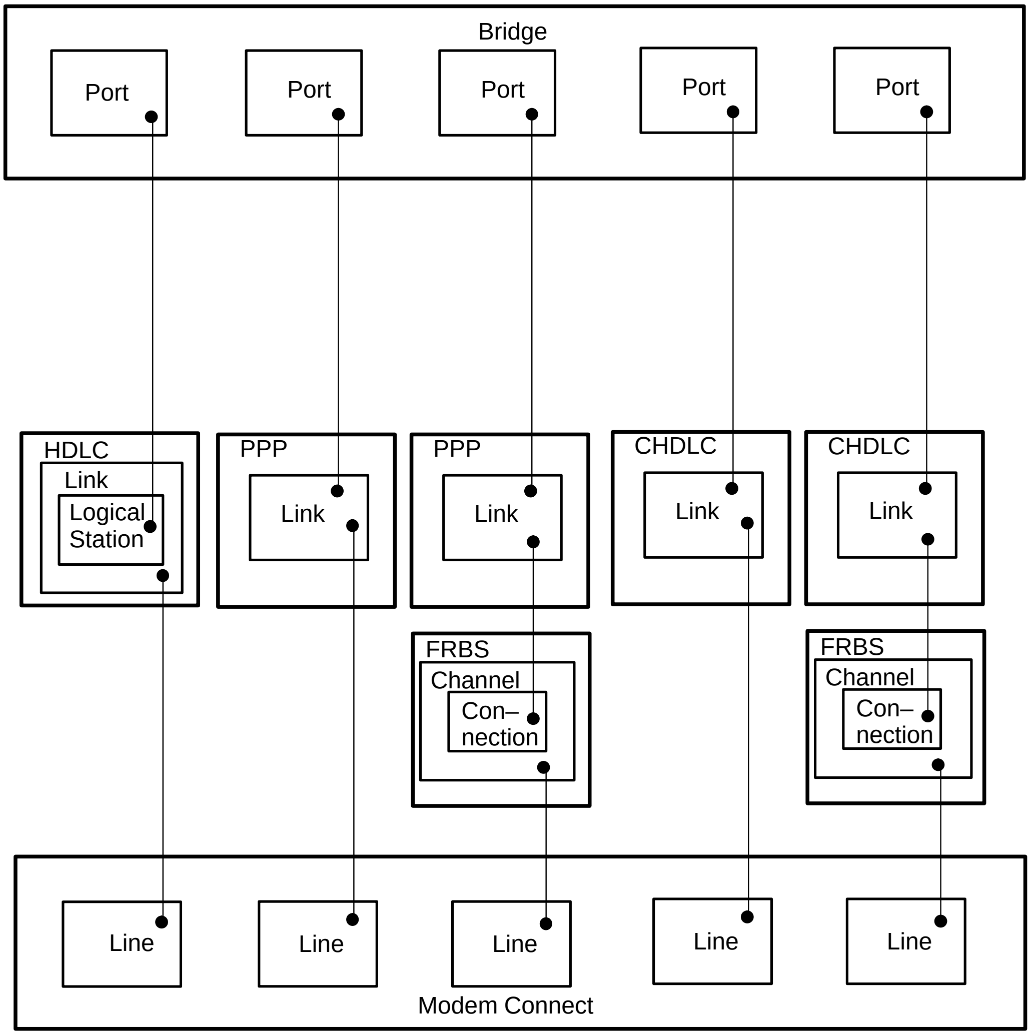


An association between two entities (indicated by a line connecting them) is made by specifying the name of the lower entity as a characteristic value of the upper entity.

For example: a bridge port of type FDDI specifies as its DATA LINK entity the name of the FDDI Station Link.

CBN–0073–93–I

Figure 21–3 Entities Required by Remote PPP, HDLC, and CHDLC Bridge Ports

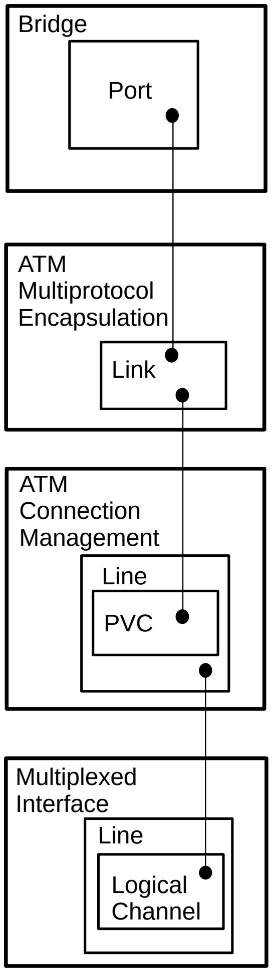


An association between two entities (indicated by a line connecting them) is made by specifying the name of the lower entity as a characteristic value of the upper entity.

For example: a bridge port of type HDLC specifies as its DATA LINK entity the name of the HDLC Logical Station.

CBN–0074–93–I

Figure 21–4 Entities Required by Remote ATM Permanent Bridge Ports



An association between two entities (indicated by a line connecting them) is made by specifying the name of the lower entity as a characteristic value of the upper entity.

For example: a bridge port of type ATM specifies as its DATA LINK entity the name of the ATM Multiprotocol Encapsulation Link.

CBN–0002–95–I

21.4 Routing and Bridging of Protocols

21.4.1 Protocols that Will Not Be Bridged

The following protocols will not be bridged by the DECNIS, irrespective of how routing is configured, or even if there are no routing circuits:

- Any OSI routing protocols
- Any DECnet Phase IV routing protocols

21.4.2 Protocols that Can Be Bridged or Routed

The DECNIS will bridge or route the following protocols, depending on its configuration:

- IP
- AppleTalk
- NetWare IPX

21.4.3 How the DECNIS Decides How to Treat These Protocols

If routing of the above protocols is enabled on the DECNIS, then they will be routed, rather than bridged.

If routing of the above protocols is disabled, then they will be bridged, by default.

21.4.4 Switching Between Routing and Bridging

To alter the way the DECNIS treats these protocols, run the DECNIS text-based configurator and select the Configuration Options section. This section allows you to choose whether the DECNIS will route IP, AppleTalk, and NetWare IPX.

Select No if you want the DECNIS to bridge these protocols (provided that bridging is enabled on the DECNIS).

21.4.5 Implications for SNMP

If you choose not to include IP in the routing protocols, you will not be able to use SNMP to monitor the DECNIS.

21.4.6 Protocol Filtering

If these protocols are not routed, you can still prevent them from being bridged by setting up bridge filtering: see Chapter 22.

21.5 Setting Up the DECNIS as a Bridge

21.5.1 Introduction

This section describes how to configure the DECNIS so that it operates as a bridge, joining two or more LAN segments. It describes how to set up both local and remote bridge ports.

21.5.2 DECNIS Requirements

Number of Bridge Ports

The DECNIS software allows you to configure up to 15 bridge ports.

Remote Bridge Ports: Network Interface Cards

You can only set up a remote bridge port on the Network Interface Cards listed in Table 21–1:

Table 21–1 Remote Bridge Ports: Network Interface Cards and Data Links

Network Interface Cards Supported	Data Links Supported on Each Card
DEC WANcontroller 614/618	HDLC, PPP or CHDLC
DEC WANcontroller 622	HDLC, PPP, CHDLC or VCP
DECNIS ATMcontroller 631	ATM Permanent

See Section 21.5.3 for the procedure to set up a bridge connection. Refer to the following sections for descriptions of how to set up the initial connections:

- Section 8.6 describes how to set up an ATM connection.
- Section 8.19 describes how to set up a CHDLC connection.
- Section 8.10 describes how to set up an HDLC connection.
- Section 8.18 describes how to set up a PPP connection.
- Section 8.9 describes how to set up a VCP connection.

Remote Bridge Ports: Line Speeds

Remote bridge ports should not be run at line speeds of less than 56 kbits/s.

21.5.3 Procedure

Follow these steps to set up bridging on the DECNIS.

1. Create the BRIDGE entity:

   ```
   NCL> CREATE BRIDGE
   ```

2. Enable the BRIDGE entity:

   ```
   NCL> ENABLE BRIDGE
   ```

3. Create the PORT entity for the relevant connection.

   ```
   NCL> CREATE BRIDGE PORT port-name PORT NUMBER n, TYPE type
   ```

   where:

n	is a decimal number between 1 and 15. Each bridge port on the DECNIS must have a unique number.
port-name	is a name you can use to identify the port. You can set this to be the same as the port number.
type	is one of LOCAL, REMOTE, or VCP.

4. Associate the data link for the connection with the port.

   - For a local bridge port on a CSMA/CD connection, and a remote bridge port on a VCP connection:

     ```
     NCL> SET BRIDGE PORT port-name -
     _NCL> DATA LINK ENTITY CSMA-CD STATION station-name
     ```

     where *station-name* is the name of the CSMA/CD station.

   - For a local bridge port on an FDDI connection:

     ```
     NCL> SET BRIDGE PORT port-name -
     _NCL> DATA LINK ENTITY FDDI STATION station-name
     ```

     where *station-name* is the name of the FDDI station.

   - For a remote bridge port on an HDLC connection:

     ```
     NCL> SET BRIDGE PORT port-name DATA LINK ENTITY -
     _NCL> HDLC LINK link-name LOGICAL STATION station-name
     ```

     where *station-name* is the name of the HDLC logical station.

   - For a remote bridge port on a PPP connection:

     ```
     NCL> SET BRIDGE PORT port-name DATA LINK ENTITY -
     _NCL> PPP LINK link-name
     ```

     where *link-name* is the name of the PPP link.

- For a remote bridge port on an ATM connection:

  ```
  NCL> SET BRIDGE PORT port-name DATA LINK ENTITY -
  _NCL> ATM MULTIPROTOCOL ENCAPSULATION LINK link-name
  ```

  where *link-name* is the name of the ATM Multiprotocol Encapsulation link.

- For a remote bridge port on a CHDLC connection:

  ```
  NCL> SET BRIDGE PORT port-name DATA LINK ENTITY -
  _NCL> CHDLC LINK link-name
  ```

  where *link-name* is the name of the CHDLC link.

5. Enable the port:

   ```
   NCL> ENABLE BRIDGE PORT port-name
   ```

6. Repeat steps 3 to 5 for each connection you wish to bridge.

21.5.4 Checking the MANUAL DATA LINK SDU SIZE

Note the following:

- The value of the MANUAL DATA LINK SDU SIZE for a CHDLC bridge port must be less than or equal to the MAXIMUM RECEIVE SDU SIZE on the CHDLC LINK specified as the DATA LINK ENTITY.
- The value of the MANUAL DATA LINK SDU SIZE for a PPP bridge port must be less than or equal to the PREFERRED MAXIMUM RECEIVE SDU SIZE on the PPP LINK specified as the DATA LINK ENTITY.

21.6 Using VCP Bridging

Note the following if you set up bridge ports on VCP connections:

- If you have parallel bridge links between the DECNIS and the Vitalink system, only one link will be active at a time.
- DLS (Distributed Load Sharing) is not supported by the DECNIS. DLS is a Vitalink algorithm that allows the use of what would normally be redundant links in the wide area spanning tree network.
- The Vitalink system must be set up to send and receive NV frames on synchronous lines (Vitalink does this by default).
- If the Vitalink system is used in Frame Compression mode, all frames must be compressed (not just LAT frames). Both the DECNIS and Vitalink system must have compression enabled.

21.7 Using NCL to Monitor a Bridge Port

You can use the following commands to check how a bridge port is functioning:

- To check the state of a port:

  ```
  NCL> SHOW BRIDGE PORT port-name PORT STATE
  ```

  This shows the state in which the spanning tree algorithm has placed the port. If the spanning tree algorithm has been disabled, the state should be FORWARDING.

- To check how a port is being used:

  ```
  NCL> SHOW BRIDGE PORT port-name ALL COUNTERS
  ```

  See the NCL online help for a description of the counters displayed.

21.8 Deleting a Bridge Port

21.8.1 Introduction

Delete a bridge port if you want to prevent permanently the DECNIS from forwarding or receiving frames on that port.

Use the DISABLE command to prevent temporarily the port from being used to receive or forward frames.

21.8.2 Procedure

1. Disable the bridge port:

   ```
   NCL> DISABLE BRIDGE PORT port-name
   ```

2. Remove the port name from the PORT SET characteristic of the following entities (see Section 22.14):

   - STATIC PHYSICAL ADDRESS
   - STATIC MULTICAST ADDRESS
   - FILTER SAP
   - FILTER PID
   - FILTER TYPE

   Also ensure that the port is not specified as the MANUAL DESTINATION PORT of any STATIC PHYSICAL ADDRESS entities (see Section 22.15).

3. Delete the bridge port:

```
NCL> DELETE BRIDGE PORT port-name
```

21.9 The Spanning Tree Algorithm

21.9.1 Definition

A **Spanning Tree Algorithm** is a program that the bridges in an extended LAN run in order to achieve:

- Loop avoidance. If bridges are configured in a loop, the algorithm computes a loop-free portion of the topology.
- Automatic redundancy. The algorithm detects redundant bridges, and disables their use. They will automatically be reenabled when required.

The spanning tree algorithm determines which bridges become the root bridge and the designated bridges for an extended LAN.

21.9.2 Bridge Hellos

The bridges in an extended LAN communicate by exchanging Hello messages among themselves. They use the information contained in these Hello messages to compute the spanning tree algorithm.

21.9.3 Different Modes of the Spanning Tree Algorithm

The DECNIS can operate the LAN Bridge 100 or the IEEE 802.1d spanning tree algorithm:

- IEEE 802.1d Mode

  If the DECNIS is in IEEE 802.1d mode, it will use the IEEE 802.1d implementation of the spanning tree algorithm.

- Autoselect Mode

  If the DECNIS is in autoselect mode, it will use the LAN Bridge 100 implementation of the spanning tree algorithm, provided there is another bridge in the extended LAN using the LAN Bridge 100 algorithm.

  If all the other bridges in the extended LAN use the IEEE 802.1d algorithm, the DECNIS will continue to use the IEEE 802.1d algorithm.

Restriction

The two modes of spanning tree algorithm use incompatible Hello messages. Therefore, only one mode should be used throughout any extended LAN.

21.10 Changing the Spanning Tree Algorithm on the DECNIS

21.10.1 Introduction

This section describes how to:

- Disable the spanning tree algorithm on a particular port.
- Switch between the IEEE 802.1d mode and the LAN Bridge 100 mode of the spanning tree algorithm.

21.10.2 Switching Between Spanning Tree Algorithm Modes

Set the LB100 SPANNING TREE COMPATIBILITY characteristic of the BRIDGE entity:

```
NCL> SET BRIDGE LB100 SPANNING TREE COMPATIBILITY mode
```

where *mode* is either AUTO SELECT or IEEE 802.1d.

Warning

The default mode is AUTO SELECT. If you change it to IEEE 802.1d, you must not have any bridges on the extended LAN that issue LAN Bridge 100 Hello messages; otherwise two spanning trees will be set up on the extended LAN, leading to instability and unpredictability.

21.10.3 Disabling the Spanning Tree Algorithm

You should consider disabling the spanning tree algorithm on any remote bridge ports.

If you do not disable the spanning tree algorithm on a remote bridge port, delays can result while the LAN segment connected by the bridge port reacts to spanning tree topology changes. However, if you have more than one remote bridge port in your extended LAN, you may need to operate the spanning tree algorithm to prevent loops and detect redundancy.

Procedure

Issue this command to disable the spanning tree algorithm on a bridge port:

```
NCL> SET BRIDGE PORT port-name SPANNING TREE ENABLED FALSE
```

where *port-name* is the name of the remote bridge port.

21.11 Root Bridge and Designated Bridges

21.11.1 Root Bridge

Each extended LAN has a unique root bridge. The root bridge initiates Hello messages that propagate to other bridges on the extended LAN, and dictates the values of certain spanning tree parameters for other bridges.

The port on which a bridge receives the root bridge Hello message is known as the bridge's inlink.

21.11.2 Designated Bridges

Every LAN in an extended LAN has a designated bridge. This is defined as the bridge that logically connects the LAN to the next LAN closer to the root bridge.

A bridge can be the designated bridge on any of its ports, except its inlink.

You can influence whether a bridge becomes the designated bridge for a LAN by adjusting the path cost from the bridge to the root bridge: the bridge with the lowest path cost to the root bridge is the designated bridge for that LAN.

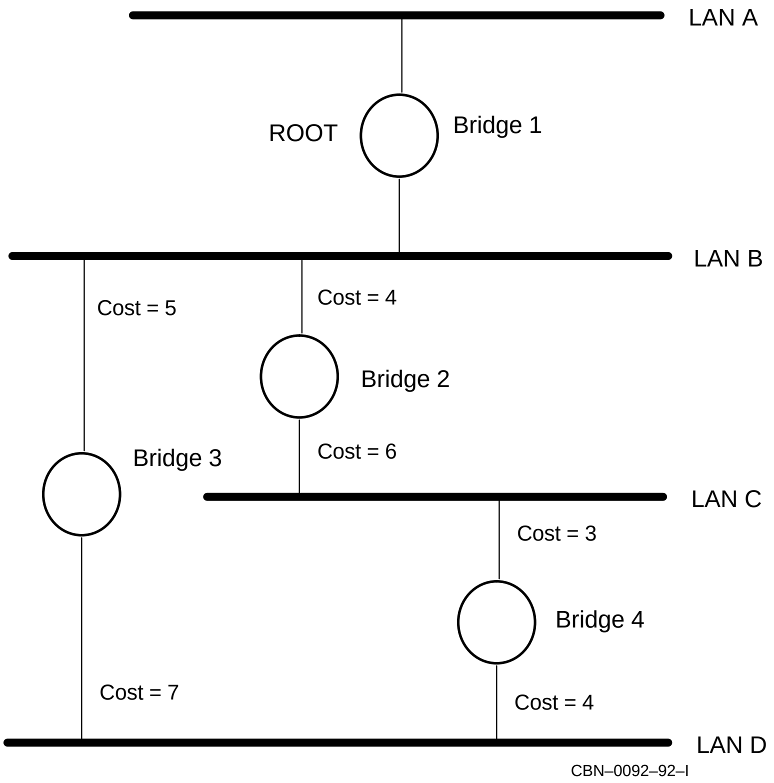
21.11.3 Example

Figure 21–5 illustrates how port cost determines the route that a frame takes through an extended LAN, and which bridges are redundant.

The path cost to the root for a LAN is the sum of all the port costs on the ports of the bridges connecting the LAN to the root bridge. The bridge with the lowest path cost to the root bridge is the designated bridge for that LAN. The port used to connect to the root bridge is the designated port's inlink.

Therefore, the designated bridge for LAN D is Bridge 3, at a cost of 5, rather than Bridge 4 at a cost of 3 + 4 = 7. Bridge 4 is therefore placed in backup mode by the spanning tree algorithm.

Figure 21–5 Extended LAN Showing Path Costs



21.12 Changing Bridge Priority

21.12.1 Introduction

In an extended LAN, the bridge with the lowest priority becomes the root bridge.

21.12.2 Procedure

Set the ROOT PRIORITY characteristic of the BRIDGE entity using the following command:

```
NCL> SET BRIDGE ROOT PRIORITY n
```

where *n* is a decimal number between 0 and 255. The lower the value of *n*, the more likely that the DECNIS will be the root bridge.

21.13 Changing the Port Cost

21.13.1 Introduction

Each port of every bridge on an extended LAN has an associated port cost. This cost is used to control how frames travel through the extended LAN.

21.13.2 Procedure

Change the port cost of a bridge port using the following command:

```
NCL> SET BRIDGE PORT port-name PORT COST n
```

where *n* is a decimal number between 0 and 255.

21.14 Minimum-Sized Ethernet/802.3 Frame Compression

21.14.1 Definition

Minimum-Sized Ethernet Frame Compression is a procedure for compressing minimum-sized Ethernet frames sent over PPP and VCP bridge ports. It is only available over PPP and VCP bridge ports.

Defined by RFC 1220 as **Tinygram Compression**, it can be used to reduce the amount of data sent over the bridge, increasing the speed of data transfer.

21.14.2 When to Use Frame Compression

You can use frame compression if all of the following apply:

- You are bridging a protocol that uses minimum-sized Ethernet frames, for example, LAT.
- You are bridging over a PPP or VCP bridge port. Section 21.5 describes how to set up PPP and VCP bridge ports.
- The line has a speed and packet rate of:
  - 1.026 Mbits/s maximum at 4400 packets/s
  - 2.048 Mbits/s maximum at *less* than 4400 packets/s
- The remote system has enabled Tinygram Compression.
- On a VCP bridge port, the Vitalink system at the remote end must be set to compress all frames (not just LAT frames).

21.14.3 How Frame Compression Works

A minimum-sized Ethernet frame can contain both user data and extra bytes of data added to make it up to the required minimum size. The extra data consists of one or more zeros included in the **real data** field of a frame. This field contains all required data except the Cyclic Redundancy Check (CRC).

The frame is compressed by removing the zeros from the real data field so that the CRC at the end of the frame remains unchanged. The frame is then sent across the bridge in its compressed form. When it reaches the destination node, zeros are added to the real data field again and the frame is restored to its original size.

21.14.4 Procedure: PPP Bridge Ports

Enter the following commands:

```
NCL> DISABLE PPP LINK link-name

NCL> SET PPP LINK link-name PREFER BRIDGE TINYGRAM COMPRESSION TRUE

NCL> ENABLE PPP LINK link-name
```

where *link-name* is the name of the PPP link.

21.14.5 Procedure: VCP Bridge Ports

Enter the following commands:

```
NCL> DISABLE CSMA-CD STATION station-name
NCL> SET CSMA-CD STATION station-name TINYGRAM COMPRESSION TRUE
NCL> ENABLE CSMA-CD STATION station-name
```

where *station-name* is the name of the CSMA/CD station.

21.15 Bridging and DECnet Phase IV Addresses

21.15.1 Problems with External Bridges and DECnet Phase IV Addresses

If the DECNIS has been given a Phase IV address (see Section 16.2), and it has any LAN routing circuits, you may not be able to put a bridge in parallel with it.

Figure 21–6 illustrates the situation.

The station address of the DECNIS will be the same for each of these LANs. This address will be AA-00-04-00-*XX-XX*, where *XX-XX* is derived from the Phase IV prefix of the DECNIS, as described in Section 16.3.

If you try to bridge these LANs using an external (non-DECNIS) bridge, the result will be an extended LAN with two stations having the same address. This is a misconfiguration.

21.15.2 Solutions

If you need to use the configuration shown in the above example, do one of the following:

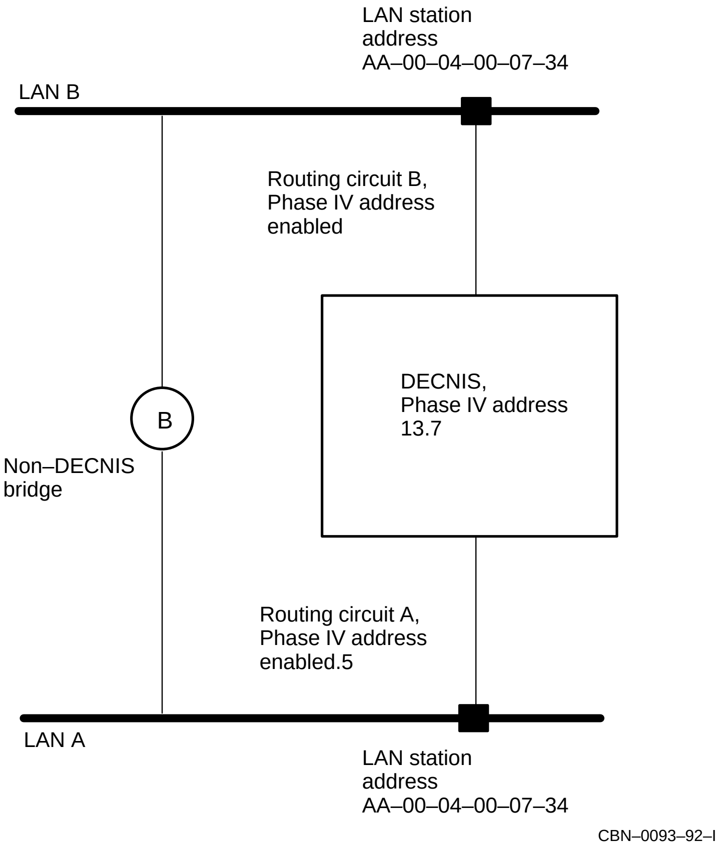
- Replace the external bridge shown in Figure 21–6 by a DECNIS operating as a bridge, or use the bridging function of the DECNIS instead of an external bridge.

  The DECNIS, when operating as a bridge, does not learn addresses from routing traffic.

- Set the ENABLE PHASEIV ADDRESS characteristic of one of the routing circuits to FALSE:

  ```
  NCL> SET ROUTING CIRCUIT circuit-name ENABLE PHASEIV ADDRESS FALSE
  ```

Figure 21–6 Illegal Routing and Bridging Between Two LANs



This will prevent the LAN station address on this circuit from becoming the Phase IV style LAN address.

21.16 Remote Bridging and Loopback

You should not use loopback testing at the physical level on HDLC, PPP, or ATM Permanent data links that are being used for remote bridging. That is:

- for HDLC and PPP links you should not switch the modem to operate in loopback mode, nor should you issue the LOOPBACK command on the relevant MODEM CONNECT LINE entity.

- for ATM Permanent links you should not switch the multiplexed interface to operate in loopback mode, nor should you issue the LOOPBACK command on the relevant MULTIPLEXED INTERFACE LOGICAL CHANNEL entity.

Using loopback on remote bridge ports will result in unpredictability and instability on the extended LAN. If the spanning tree algorithm is enabled on the remote bridge port, such problems will be relatively short-lived; if the spanning tree algorithm is disabled, management intervention will be required to solve the problem.

Note that PPP and ATM Permanent data links do not report the fact that the line is in loopback mode.

21.17 Using SNMP to Manage Bridging

The following sections provide information on the DECNIS implementation of some MIB variables related to the bridging function.

dot1dStpPortEnable: Enabling and Disabling Bridge Ports

MIB: Bridge MIB, *MIB group:* dot1dStp, *MIB table:* dot1dStpPortTable

Set this to "enabled" to enable the bridge port; set it to "disabled" to disable it.

dot1dStaticTable

MIB: Bridge MIB, *MIB group:* dot1dStatic, *MIB table:* dot1dStaticTable

The DECNIS implements the variables in this table as read-only. These are:

dot1dStaticAddress
dot1dStaticReceivePort
dot1dStaticAllowedToGoTo
dot1dStaticStatus

ebrChar MIB Group

MIB: DEC Vendor MIB elanext V2.7, *MIB group:* ebrChar

The DECNIS implements the following variables as read-only:

ebrManualFilterSwitch
ebrFragmentationSwitch
ebrRemoveMgmtAddress
ebrRemoveMgmtProto

22

Bridge Filtering on the DECNIS

22.1 Introduction

22.1.1 General

This chapter describes how to configure the bridging functions of the DECNIS to filter or forward frames based on:

- Source or destination address
- Frame format
- Protocol identifier
- DECNIS bridge port that receives or transmits the frame

22.1.2 NCL Commands

All tasks performed using NCL commands assume that you have:

- Logged on to a suitable host system or logged on to a console terminal.
- Started NCL and, if managing a remote DECNIS, set default to the DECNIS as described in Section 1.4.4 or Section 2.13.5

22.1.3 Entering Bridge Filtering Commands in the User NCL Script Files

To set up bridge filtering using the user NCL script files, you must place all filtering commands, including CREATE and SET commands, in the NIS_mop-client-name_EXTRA_ENABLE.NCL file.

Example

To prevent the DECNIS from forwarding frames bearing the source address 02-02-02-02-02-02, place the following commands in the NIS_mop-client-name_EXTRA_ENABLE.NCL file:

```
CREATE BRIDGE STATIC PHYSICAL ADDRESS 02-02-02-02-02-02
SET BRIDGE STATIC PHYSICAL ADDRESS 02-02-02-02-02-02 PORT SET { }
```

This applies to any commands involving one of the following entities:

STATIC PHYSICAL ADDRESS
STATIC MULTICAST ADDRESS
FILTER TYPE
FILTER SAP
FILTER PID

22.2 Bridge Filtering

22.2.1 Definition

When a bridge port filters a frame, it does not attempt to forward it to its destination LAN segment.

The DECNIS can filter frames based on the following attributes of the frame:

- Source address
- Destination address
- Protocol type

22.2.2 Filtering by Address

A bridge on an extended LAN has a list, for each of its ports, of the addresses that it can reach using that port. This list is maintained in two ways:

- Automatically. As the bridge receives frames, it notes their addresses and the ports on which they were received. In this way, it learns whether a particular frame should be forwarded, and the port on which it should be forwarded.
- Manually. For example, you can issue network management commands to specify that frames bearing certain source or destination addresses should only be forwarded on certain ports.

22.2.3 Filtering by Protocol Type

The bridge may have a list, for each of its ports, of the protocols that it can receive and transmit on that port. This information is not learned automatically by the bridge, but must be entered using network management commands.

22.3 Frame and Protocol Types

22.3.1 Frame Types

Each frame on the extended LAN is in one of three formats:

- Ethernet format
- IEEE 802.2 normal format
- 802.2 SNAP SAP format (PID)

22.3.2 Protocol Types

The protocol type of a particular frame is determined by its protocol identifier field.

22.3.3 Examples of Protocol Types

- An Ethernet format frame has a protocol identifier field of 2 bytes, in the form *nn-nn*.

  Ethernet frames with a protocol identifier of 08-00 are TCP/IP frames.
- An 802.2 SNAP SAP format frame has a protocol identifier field of 5 bytes, in the form *nn-nn-nn-nn-nn*.

  802.2 SNAP SAP frames with a protocol identifier of 08-00-2B-60-01 are MOP Dump/Load frames.

22.3.4 Lists of Protocol Types

Section D.1 lists some values for Ethernet protocol types.

Section D.2 lists some values for IEEE 802.2 normal format protocol types.

Section D.3 lists some values for 802.2 SNAP SAP protocol types.

22.4 DECNIS Filtering Process

22.4.1 Introduction

Frames arriving at a particular bridge port are filtered or forwarded depending on the configuration of the DECNIS.

If the frame is accepted, as shown in Figure 22–1, the same checks are then carried out at the bridge port on which the frame is to be forwarded.

22.4.2 Flowchart: Bridge Filtering

Figure 22–1 illustrates bridge filtering on the DECNIS.

22.5 Management Entities Used for Protocol Filtering

22.5.1 Introduction

This section lists the entites used for controlling how the DECNIS filters or forwards frames with a particular protocol identifier.

22.5.2 Filtering Entities

Table 22–1 shows which entity is responsible for filtering which frame types, and the corresponding bridge port characteristic.

If the relevant bridge port characteristic is set to FILTER, no frames will be forwarded by the port unless the entity exists and specifies the port concerned.

Figure 22–1 Bridge Filtering on the DECNIS

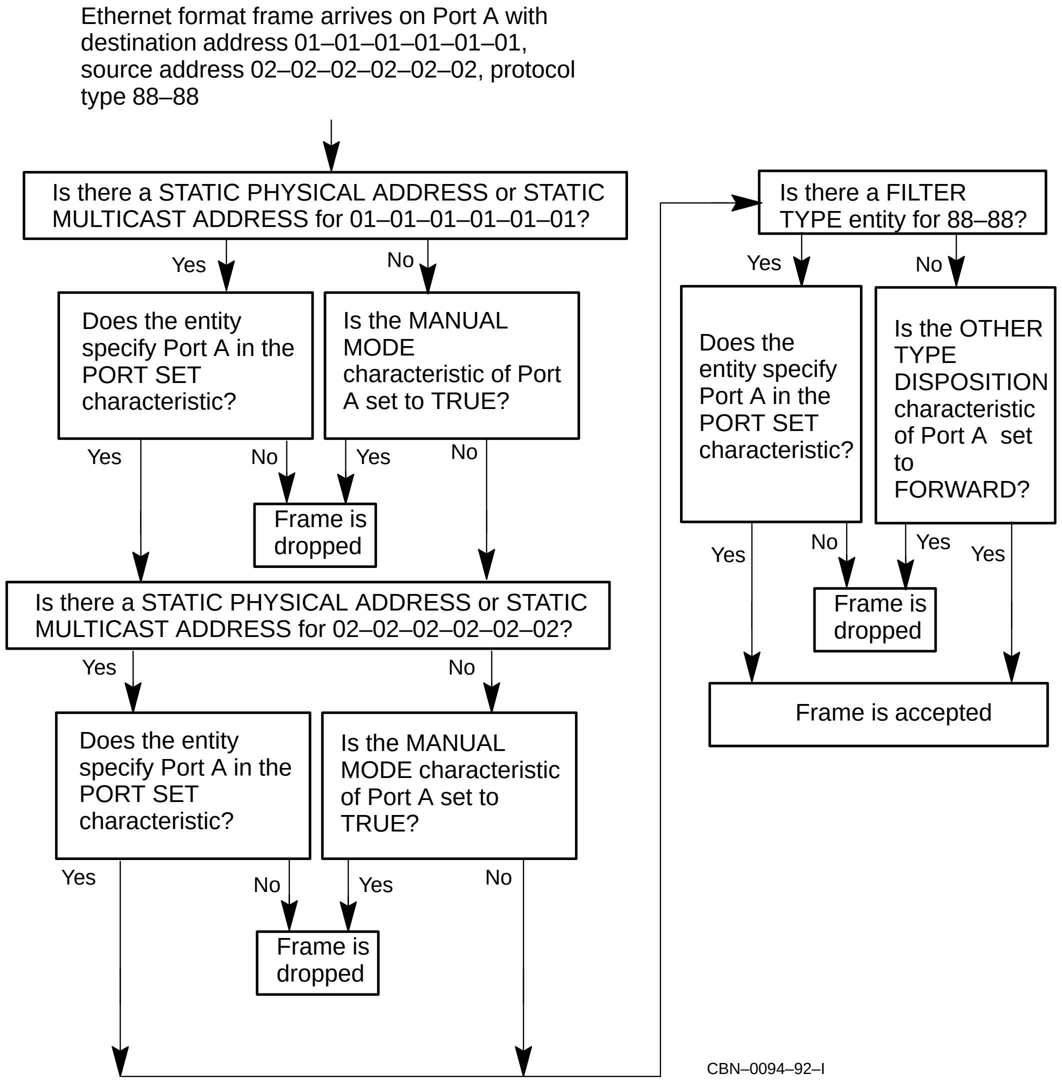

Ethernet format frame arrives on Port A with destination address 01–01–01–01–01–01, source address 02–02–02–02–02–02, protocol type 88–88
Is there a STATIC PHYSICAL ADDRESS or STATIC MULTICAST ADDRESS for 01–01–01–01–01–01?
Yes
No
Does the entity specify Port A in the PORT SET characteristic?
Is the MANUAL MODE characteristic of Port A set to TRUE?
Yes
No
Yes
No
Frame is dropped
Is there a STATIC PHYSICAL ADDRESS or STATIC MULTICAST ADDRESS for 02–02–02–02–02–02?
Yes
No
Does the entity specify Port A in the PORT SET characteristic?
Is the MANUAL MODE characteristic of Port A set to TRUE?
Yes
No
Yes
No
Frame is dropped
Is there a FILTER TYPE entity for 88–88?
Yes
No
Does the entity specify Port A in the PORT SET characteristic?
Is the OTHER TYPE DISPOSITION characteristic of Port A set to FORWARD?
Yes
No
Yes
Yes
Frame is dropped
Frame is accepted
CBN–0094–92–I

Table 22–1 Bridge Filtering Entities

Filtering Entity	Bridge Port Characteristic Set to FILTER	Effect
FILTER TYPE	OTHER TYPE DISPOSITION	The port cannot forward Ethernet type frames, unless there is a FILTER TYPE entity for Ethernet frames of a particular protocol, and this entity specifies the port concerned.
FILTER SAP	OTHER SAP DISPOSITION	The port cannot forward IEEE 802.2 normal format frames, unless there is a FILTER SAP entity for 802.2 normal format frames of a particular protocol, and this entity specifies the port concerned.
FILTER PID	OTHER PID DISPOSITION	The port cannot forward 802.2 SNAP SAP format frames, unless there is a FILTER PID entity for 802.2 SNAP SAP frames of a particular protocol, and this entity specifies the port concerned.

22.6 Permitting a Port to Bridge Only Specified Addresses

22.6.1 Introduction

You can configure a bridge port so that it only bridges those addresses that you enter manually, and ignores traffic to or from other addresses.

This section describes how to prevent a bridge port on the DECNIS from using the dynamically learnt information about the physical addresses in the extended LAN that can be reached on that port.

22.6.2 Procedure

Follow these steps to force the DECNIS to bridge only those addresses and protocols you specify:

1. Set the relevant bridge port to bridge in manual mode:

   ```
   NCL> SET BRIDGE PORT port-name MANUAL MODE TRUE
   ```

Result

This forces Port *port-name* to ignore any learnt addresses.

2. You must now enter manually the addresses and protocols that this bridge port will bridge: see Sections 22.10, 22.11, 22.12, and 22.13.

22.7 Permitting a Port to Bridge Only Specified Ethernet Protocol Types

22.7.1 Introduction

This section describes how to configure a bridge port so that it only bridges those Ethernet protocols that you enter manually, and ignores other Ethernet protocol traffic.

22.7.2 Procedure

Follow these steps to force the DECNIS to bridge only those Ethernet frames you specify:

1. Set the OTHER TYPE DISPOSITION characteristic of the port to FILTER:

   ```
   NCL> SET BRIDGE PORT port-name OTHER TYPE DISPOSITION FILTER
   ```

   Result

   This ensures that Port *port-name* will not forward Ethernet frames unless there is a FILTER TYPE entity that allows the port to forward Ethernet frames of a particular protocol.

2. You must now create FILTER TYPE entities for the Ethernet protocols that this bridge port will bridge: see Section 22.11 for details of how to set up FILTER TYPE entities.

22.8 Permitting a Port to Bridge Only Specified IEEE 802.2 Normal Format Protocol Types

22.8.1 Introduction

This section describes how to configure a bridge port so that it only bridges those IEEE 802.2 normal format protocols that you enter manually, and ignores other protocol traffic using this type of frame.

22.8.2 Procedure

Follow these steps to force the DECNIS to bridge only those IEEE 802.2 normal format frames you specify:

1. Set the OTHER SAP DISPOSITION characteristic of the port to FILTER:

   ```
   NCL> SET BRIDGE PORT port-name OTHER SAP DISPOSITION FILTER
   ```

   Result

   This ensures that Port *port-name* will not forward IEEE 802.2 normal format frames unless there is a FILTER SAP entity that allows the port to forward frames of a particular protocol.

2. You must now create FILTER SAP entities for the IEEE 802.2 normal format protocols that this bridge port will bridge: see Section 22.12 for details of how to set up FILTER SAP entities.

22.9 Permitting a Port to Bridge Only Specified 802.2 SNAP SAP Protocol Types

22.9.1 Introduction

This section describes how to configure a bridge port so that it only bridges those 802.2 SNAP SAP protocols that you enter manually, and ignores other protocol traffic using this type of frame.

22.9.2 Procedure

Follow these steps to force the DECNIS to bridge only those 802.2 SNAP SAP frames you specify:

1. Set the OTHER PID DISPOSITION characteristic of the port to FILTER:

   ```
   NCL> SET BRIDGE PORT port-name OTHER PID DISPOSITION FILTER
   ```

   Result

   This ensures that Port *port-name* will not forward 802.2 SNAP SAP frames unless there is a FILTER PID entity that allows the port to forward 802.2 SNAP SAP frames of a particular protocol.

2. You must now create FILTER PID entities for the 802.2 SNAP SAP protocols that this bridge port will bridge: see Section 22.13 for details of how to set up FILTER TYPE entities.

22.10 Restricting Bridging of Certain Addresses to Specific Bridge Ports

22.10.1 Introduction

This section describes how to create STATIC PHYSICAL ADDRESS and STATIC MULTICAST ADDRESS entities. Each of these entities specifies an address of a station (or stations) on the extended LAN, and the ports that can receive and forward frames bearing this address (either as the source or destination address).

22.10.2 Procedure

Follow these steps to create STATIC PHYSICAL ADDRESS and STATIC MULTICAST ADDRESS entities:

1. For a physical address, that is, an address of a specific station on the extended LAN, create a STATIC PHYSICAL ADDRESS entity:

   ```
   NCL> CREATE BRIDGE STATIC PHYSICAL ADDRESS nn-nn-nn-nn-nn-nn
   ```

   where *nn-nn-nn-nn-nn-nn* is the physical address of a station on the extended LAN, in the form of 6 pairs of hex digits.

2. Specify the ports that can receive and forward frames with this address, either as their source or destination address: see Section 22.14.

3. For a multicast address, that is, an address used to send messages to a group of stations on the extended LAN, create a STATIC MULTICAST ADDRESS entity:

   ```
   NCL> CREATE BRIDGE STATIC MULTICAST ADDRESS nn-nn-nn-nn-nn-nn
   ```

   where *nn-nn-nn-nn-nn-nn* is a multicast address on the extended LAN, in the form of 6 pairs of hex digits.

4. Specify the ports that can receive and forward frames that have this address as their destination address: see Section 22.14.

22.11 Restricting Bridging of Certain Ethernet Protocols to Specific Bridge Ports

22.11.1 Introduction

This section describes how to create FILTER TYPE entities. Each of these entities specifies an Ethernet frame bearing a specific protocol identifier, and the ports that can receive and forward these frames.

22.11.2 Procedure

Follow these steps to create FILTER TYPE entities:

1. Create the FILTER TYPE entity as follows:

   ```
   NCL> CREATE BRIDGE FILTER TYPE nn-nn
   ```

   where *nn-nn* identifies the protocol type of the Ethernet format frame, in the form of 2 pairs of hex digits.

   Example: 60-03 specifies the DECnet protocol.

2. Specify the ports that can receive and forward Ethernet format frames that have this protocol: see Section 22.14.

22.12 Restricting Bridging of Certain IEEE 802.2 Normal Format Protocols to Specific Bridge Ports

22.12.1 Introduction

This section describes how to create FILTER SAP entities. Each of these entities specifies an IEEE 802.2 normal format frame bearing a specific protocol identifier, and the ports that can receive and forward these frames.

22.12.2 Procedure

Follow these steps to create FILTER SAP entities:

1. Create the FILTER SAP entity as follows:

   ```
   NCL> CREATE BRIDGE FILTER SAP nn
   ```

   where *nn* identifies the protocol type of the IEEE 802.2 normal format frame, in the form of a pair of hex digits.

   Example: 7E specifies the X.25 Level 3 Over ISO 8802.2 LLC2 Class 2 protocol.

2. Specify the ports that can receive and forward 802.2 normal format frames that have this protocol: see Section 22.14.

22.13 Restricting Bridging of Certain SNAP SAP Protocols to Specific Bridge Ports

22.13.1 Introduction

This section describes how to create FILTER PID entities. Each of these entities specifies a SNAP SAP frame bearing a specific protocol identifier, and the ports that can receive and forward these frames.

22.13.2 Procedure

Follow these steps to create FILTER PID entities:

1. Create the FILTER PID entity as follows:

   ```
   NCL> CREATE BRIDGE FILTER PID nn-nn-nn-nn-nn
   ```

   where *nn-nn-nn-nn-nn* identifies the protocol type of the 802.2 SNAP SAP format frame, in the form of 5 pairs of hex digits. The first 3 pairs are the identifier assigned by the IEEE to the organization that controls the protocol.

   Example: 08-00-2B is Digital's organization identifier, and 08-00-2B-80-3C specifies the DNA Naming Service protocol.

2. Specify the ports that can receive and forward 802.2 SNAP SAP format frames that have this protocol: see Section 22.14.

22.14 Specifying Ports for Filtering

22.14.1 Introduction

For each of the filtering entities, you must specify those bridge ports that can receive and transmit frames. This section describes how to specify the ports for each of these entities.

By default, no bridge ports can transmit or receive the frames specified in any particular entity.

22.14.2 PORT SET Characteristic

The PORT SET characteristic of the above entities controls which bridge ports on the DECNIS can receive and transmit the frames specified in the entities.

22.14.3 Use of the BLOCK and SET Command

BLOCK Command

Use the BLOCK command to include all bridge ports except those specified in the command.

SET Command

Use the SET command to include only the bridge ports specified in the command.

22.14.4 Command Format

The command format is as follows:

```
NCL> COMMAND BRIDGE entity PORT SET {port-name1, port-name2,...}
```

Variable	Possible Values	Meaning
COMMAND	BLOCK	The bridge ports specified in the command are to be prevented from receiving or transmitting the frames specified in the entity. This command is equivalent to using the SET command and specifying all bridge ports except those specified in the BLOCK command.

Variable	Possible Values	Meaning
	SET	The bridge ports specified in the command are to be allowed to receive or transmit the frames specified in the entity. This command is equivalent to using the BLOCK command and specifying all bridge ports except those specified in the SET command.
entity	STATIC PHYSICAL ADDRESS *nn-nn-nn-nn-nn-nn* STATIC MULTICAST ADDRESS *nn-nn-nn-nn-nn-nn* FILTER TYPE *nn-nn* FILTER SAP *nn* FILTER PID *nn-nn-nn-nn-nn*	Specifies the filtering entity with which the port is associated.
port-name1 *port-name2*		The names of the bridge ports on the DECNIS to which the command applies.

22.14.5 Requirement

You should ensure that at least two bridge ports can transmit and receive the frames specified in any particular entity; that is, the PORT SET characteristic should specify at least two bridge ports.

22.14.6 Examples

- If you want to prevent one or more bridge ports from receiving or transmitting the frames specified in the entity, use the BLOCK command as follows:

  ```
  NCL> BLOCK BRIDGE entity PORT SET {port-name1, port-name2,...}
  ```

- If you want all the bridge ports on the DECNIS to be able to receive and transmit the frames specified in the entity, use the BLOCK command without specifying any ports:

  ```
  NCL> BLOCK BRIDGE entity PORT SET { }
  ```

- If you want only two or more of the bridge ports on the DECNIS to be able to receive and transmit the frames specified in the entity, set the PORT SET characteristic:

  ```
  NCL> SET BRIDGE entity PORT SET {port-name1,port-name2,...}
  ```

- If you do not want any bridge ports on the DECNIS to be able to receive and transmit the frames specified in the entity, use the SET command without specifying any bridge ports:

  ```
  NCL> SET BRIDGE entity PORT SET { }
  ```

22.14.7 Checking the Bridge Ports

Use the SHOW command to check that the correct bridge ports have been specified for each entity as follows:

```
NCL> SHOW BRIDGE entity PORT SET
```

22.15 Locking a Destination Address to a Specific Bridge Port

22.15.1 Introduction

This section describes how to associate a particular bridge port with a specific destination address. In this way, you ensure that frames bearing this destination address are always forwarded on this bridge port.

22.15.2 Example

Figure 22–2 shows a bridge that has learnt that frames with a destination address of 02-02-02-02-02-02 can be reached from Port A.

However, if a station on LAN B masquerades as 02-02-02-02-02-02, the bridge may start to send frames addressed to 02-02-02-02-02-02 on Port B.

22.15.3 Procedure

1. Create a STATIC PHYSICAL ADDRESS entity for the destination address you want to lock to the port:

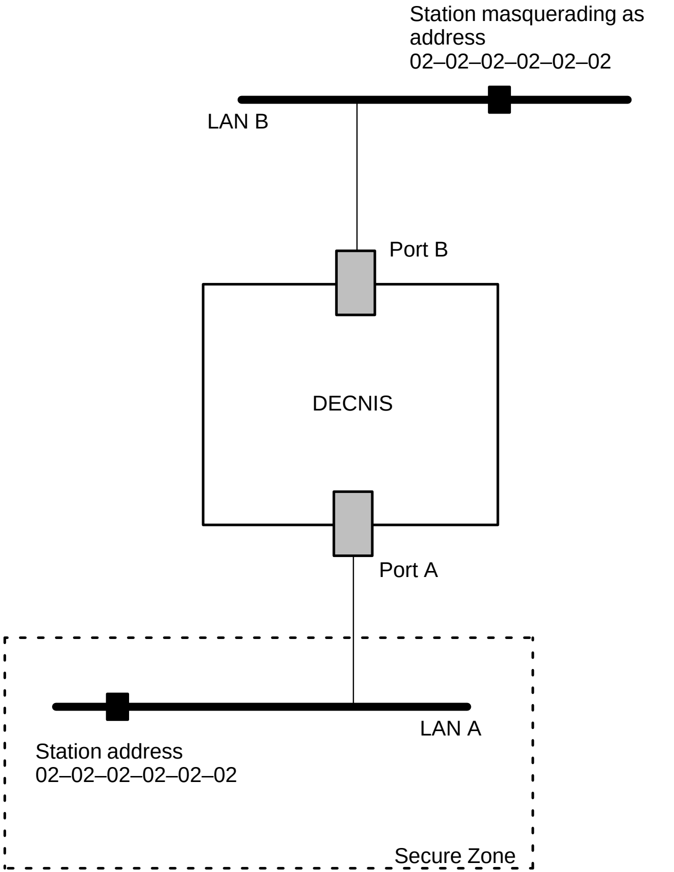
   ```
   NCL> CREATE BRIDGE STATIC PHYSICAL ADDRESS nn-nn-nn-nn-nn-nn
   ```

   where *nn-nn-nn-nn-nn-nn* is the physical address of the destination station.

2. Now specify the name of the port to which frames bearing this destination address will be forwarded:

   ```
   NCL> SET BRIDGE STATIC PHYSICAL ADDRESS nn-nn-nn-nn-nn-nn
   _NCL> MANUAL DESTINATION PORT port-name
   ```

Figure 22–2 DECNIS Operating as a Bridge Between Two LANs



where *port-name* is the name of the port on the DECNIS that you want to use to transmit frames bearing this destination address (Port A in this example).

No other port on the DECNIS will be able to transmit frames bearing this destination address.

3. Allow the other bridge ports to receive frames destined for this address:

   ```
   NCL> BLOCK BRIDGE STATIC PHYSICAL ADDRESS nn-nn-nn-nn-nn-nn -
   _NCL> PORT SET {}
   ```

   If you want a restricted set of the bridge ports to receive frames destined for this address, see Section 22.14.

22.16 Bridging of Null Internet Frames on FDDI Ports

22.16.1 Introduction

IEEE 802.2 normal format Null Internet frames have a protocol identifier of FE. The F621 Network Interface Card interprets this as a routing frame and, by default, will not forward it.

To enable forwarding of Null Internet frames over FDDI bridge ports, you must set up filtering on the relevant ports so that these frames will be forwarded.

Section 22.12.2 describes how to set up filtering for IEEE 802.2 normal format frames.

Example

Place the following commands in the NIS_*client-name*_EXTRA_ENABLE.NCL file to ensure that FDDI bridge ports FDDI-1 and FDDI-2 will always forward Null Internet frames:

```
NCL> CREATE BRIDGE FILTER SAP fe
NCL> SET BRIDGE FILTER SAP fe PORT SET {fddi-1, fddi-2}
```

22.16.2 Null Internet Frames on CSMA/CD Ports

The L601 and L602 Network Interface Cards forward Null Internet frames, by default. You can change this behavior by setting up filtering information explicitly.

22.17 Using SNMP to Manage Bridge Filtering

You can use SNMP Get requests to monitor the bridge filtering functions of the DECNIS.

You cannot use SNMP Set requests to alter its bridge filtering functions. You must use NCL commands or the DECNIS text-based configurator to set up bridge filtering. Note that the clearVISN DECNIS configurator does not support bridge filtering or bridging.

Part V

Appendixes

This part contains the following appendixes:

- Appendix A describes the management modules used by the DECNIS.
- Appendix B lists the modules and entities used by the DECNIS, together with the default values of their characteristics.
- Appendix C provides the NCL commands needed to set up an event sink for the DECNIS.
- Appendix D lists, for the commonly used protocols, the values of the protocol fields in the different types of frame format. This information is needed if using protocol filtering with bridging functions.
- Appendix E lists the values of the characteristics in the X25 ACCESS "Default" template.
- Appendix F contains tables listing requirements for connecting the DECNIS to other DECnet Phase IV and DECnet/OSI routers.
- Appendix G provides example NCL scripts, illustrating the NCL commands needed for the DECNIS to perform various functions.

A

Management Modules Used by the DECNIS

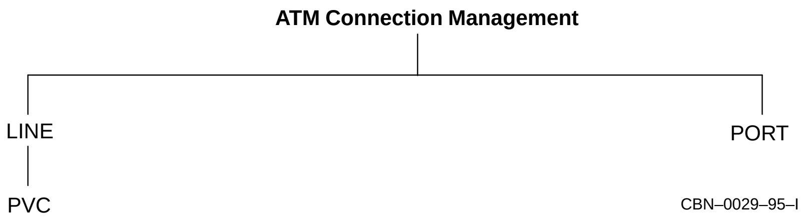
The following sections describe the DECnet/OSI modules used by the DECNIS. Your system may not use all of these modules, depending on your particular configuration.

A.1 ATM Connection Management Module

This module manages the initialization of ATM lines and the allocation and configuration of ATM PVCs.

There is a Line associated with each hardware communication port that supports the ATM protocol.

Figure A–1 ATM Connection Management Module

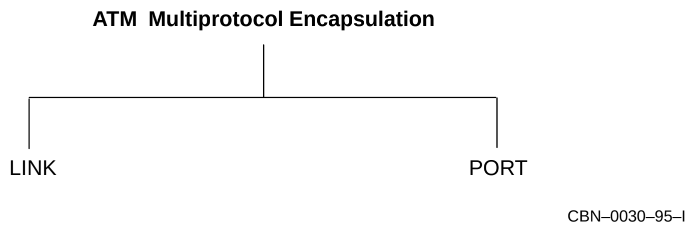


A.2 ATM Multiprotocol Encapsulation Module

This module implements multi-protocol routing and bridging over ATM.

Each Bridging port or Routing circuit must be associated with an ATM multiprotocol encapsulation Link.

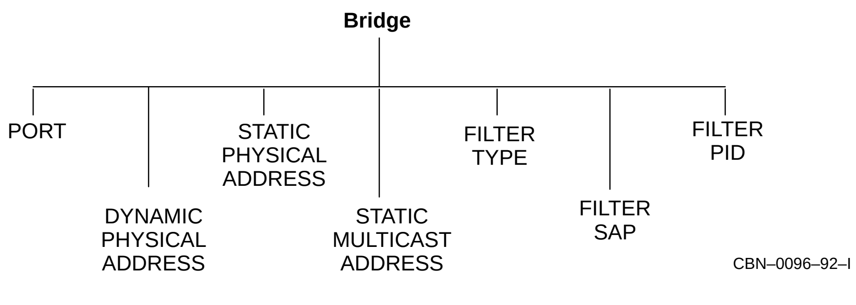
Figure A–2 ATM Multiprotocol Encapsulation Module



A.3 Bridge Module

Bridging enables stations to communicate at the Data Link level; for example, bridges can connect several LANs to form an extended LAN.

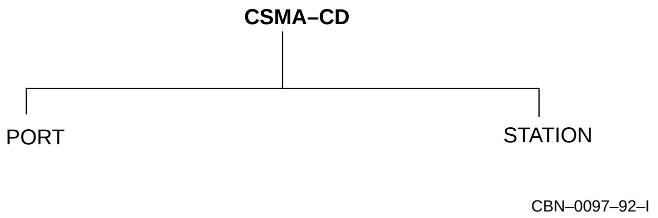
Figure A–3 Bridge Module



A.4 CSMA-CD Module

Carrier Sense, Multiple Access with Collision Detect is the Data Link protocol used by the Ethernet and ISO 8802-3 LANs.

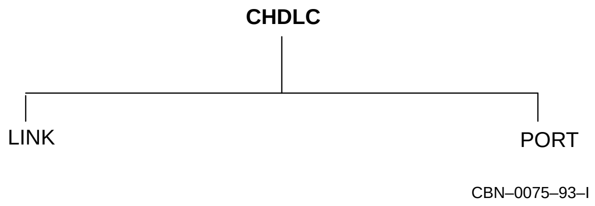
Figure A–4 CSMA-CD Module



A.5 CHDLC Module

This is a variant of the High-level Data Link Control module (see HDLC).

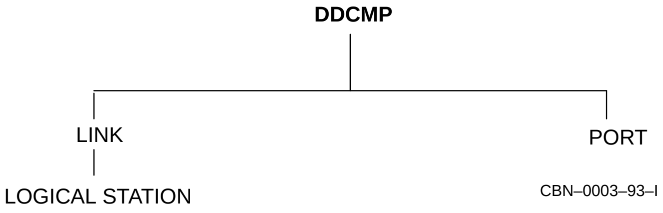
Figure A–5 CHDLC Module



A.6 DDCMP Module

The Digital Data Communications Protocol is a Digital proprietary data link protocol. The DDCMP module provides management of this protocol.

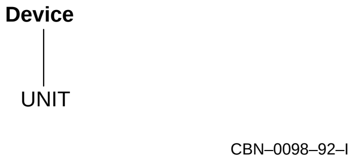
Figure A–6 DDCMP Module



A.7 Device Module

The Device module provides management of the physical devices attached to a system.

Figure A–7 Device Module



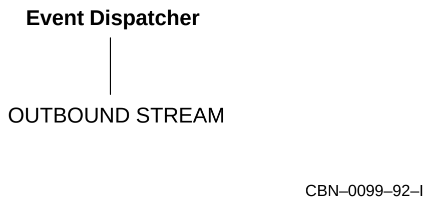
A.8 DTSS Clerk Module

The DTSS Clerk collates and synchronizes the times recorded by DTSS Servers. This module has no entities.

A.9 Event Dispatcher Module

This controls the logging of network events.

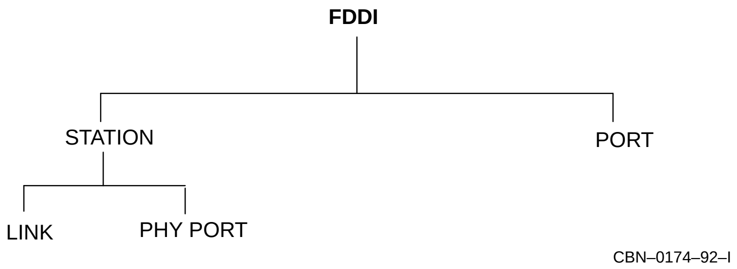
Figure A–8 Event Dispatcher Module



A.10 FDDI Module

Fiber Distributed Data Interface is the data link protocol used by fiber-optic LANs.

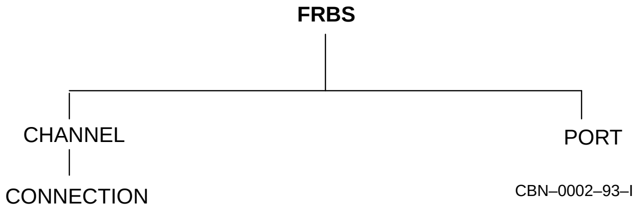
Figure A–9 FDDI Module



A.11 FRBS Module

The FRBS (Frame Relay Bearer Service) implements management of the Frame Relay functions of the DECNIS.

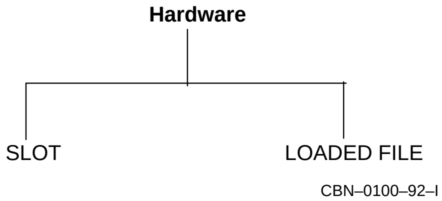
Figure A–10 FRBS Module



A.12 Hardware Module

The Hardware module provides hardware-related management of the DECNIS.

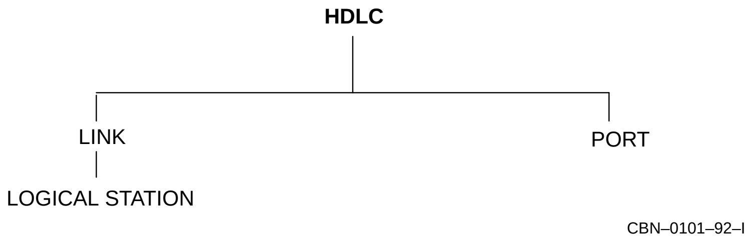
Figure A–11 Hardware Module



A.13 HDLC Module

High-level Data Link Control is the preferred protocol for providing error-free communications over low-delay, high-loss, point-to-point links.

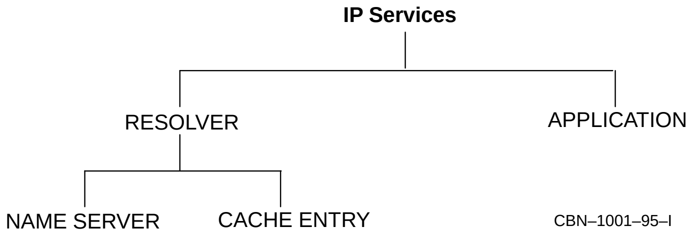
Figure A–12 HDLC Module



A.14 IP Services Module

This module manages registration and access control for applications using TCP/IP. The module also manages IP address identification.

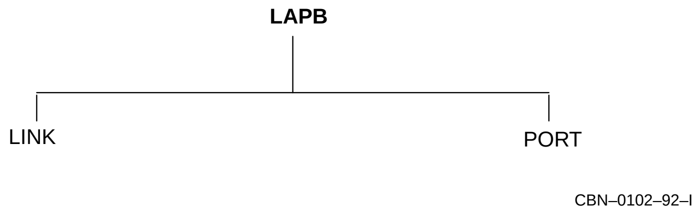
Figure A–13 IP Services Module



A.15 LAPB Module

The Link Access Procedure Balanced module defines the X.25 level 2 protocol that is used to exchange frames between a DTE and a DCE.

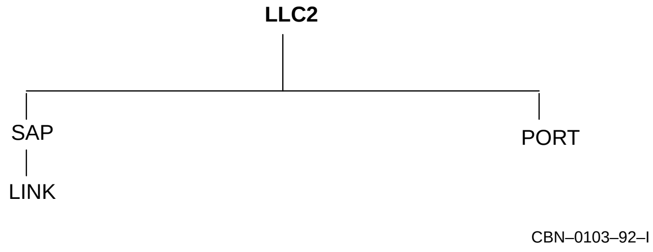
Figure A–14 LAPB Module



A.16 LLC2 Module

This module implements the LLC2 protocol that provides the ISO Connection-Oriented Network Service at the Data Link layer.

Figure A–15 LLC2 Module



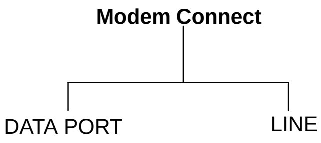
A.17 Loopback Application Module

This module controls loopback testing between two nodes at the Application layer. This module has no entities.

A.18 Modem Connect Module

This module defines the physical lines connecting two systems.

Figure A–16 Modem Connect Module

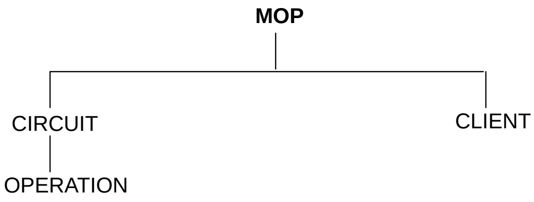


CBN–0104–92–I

A.19 MOP Module

The Maintenance Operation Protocol controls low-level communications with systems that are not fully operational, or systems that are being tested. Loading and dumping operations use MOP.

Figure A–17 MOP Module



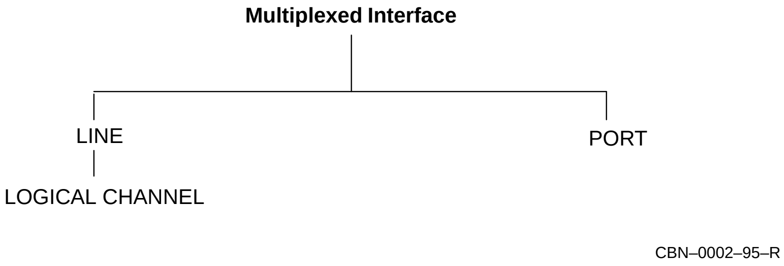
CBN–0105–92–I

A.20 Multiplexed Interface Module

This module defines physical layer characteristics for DS3/E3 and OC-3c/STM-1 devices.

There is one Logical Channel entity per Line.

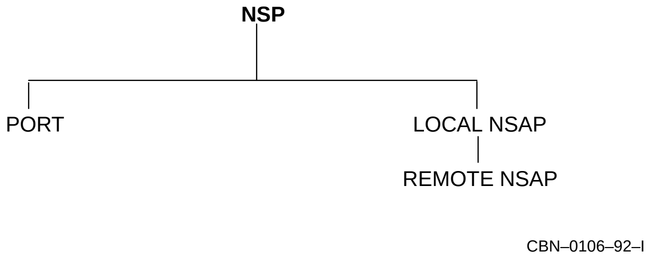
Figure A–18 Multiplexed Interface Module



A.21 NSP Module

The Network Services Protocol is responsible for the creation and deletion of communications channels, flow control, end-to-end error control, and segmentation and reassembly of messages. NSP is a proprietary DNA protocol at the Transport layer.

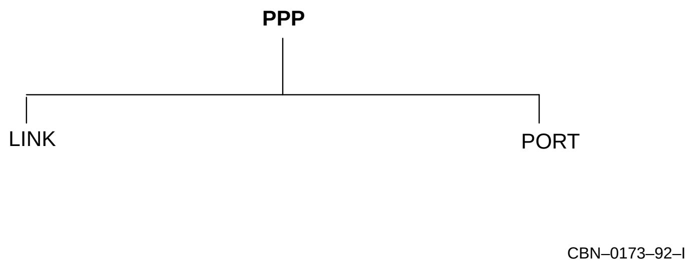
Figure A–19 NSP Module



A.22 PPP Module

The Point-to-Point Protocol is the preferred data link protocol for low-loss, high-delay links, for example, satellite links. It provides a negotiable, multiplexed facility so that packets from various network layer protocols can be transferred over the link at any one time. This allows systems from different vendors to interoperate over the link.

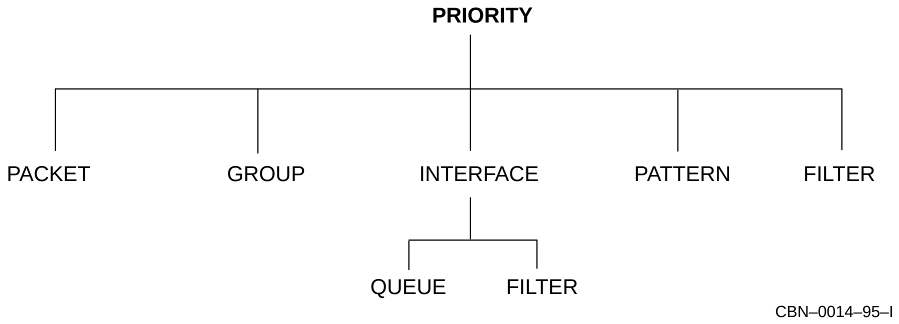
Figure A–20 PPP Module



A.23 Priority Module

The Priority module controls packet prioritization and IP packet filtering on the DECNIS.

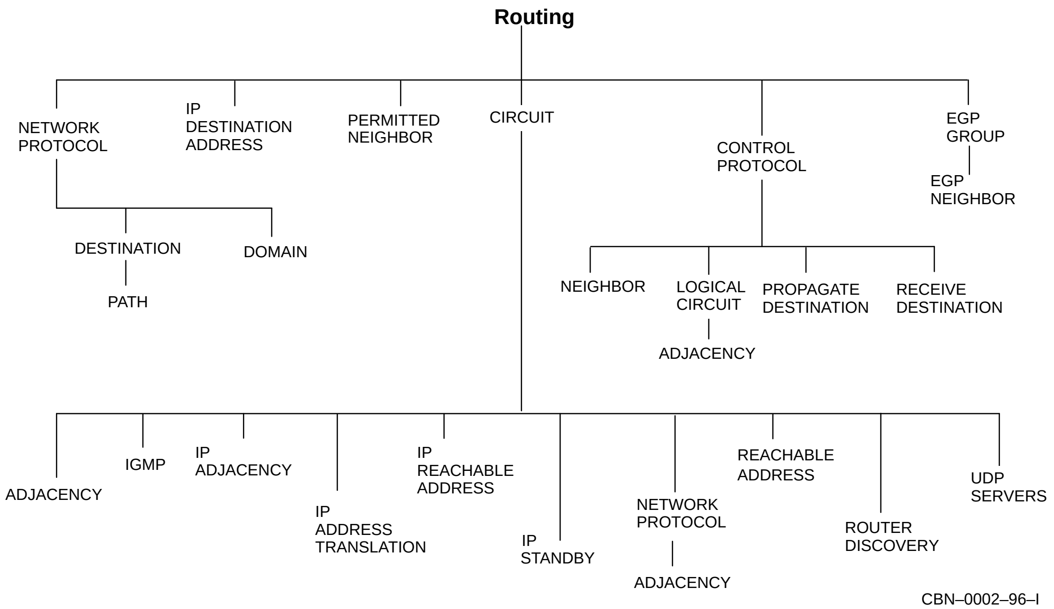
Figure A–21 Priority Module



A.24 Routing Module

This module is responsible for providing the Network layer functions. It is compatible with ISO 8473, ISO 9542, and ISO 10589, and the relevant TCP/IP RFCs.

Figure A–22 Routing Module

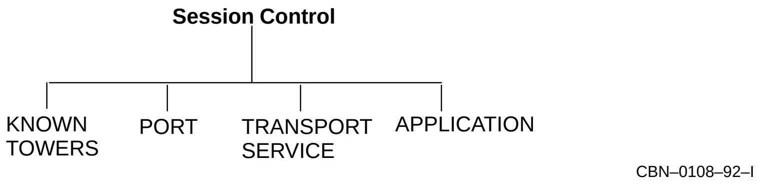


CBN–0002–96–I

A.25 Session Control Module

This defines the applications that receive incoming calls and make outgoing calls.

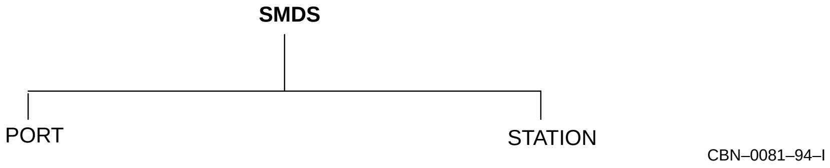
Figure A–23 Session Control Module



A.26 SMDS Module

The SMDS module controls SMDS communication on the DECNIS.

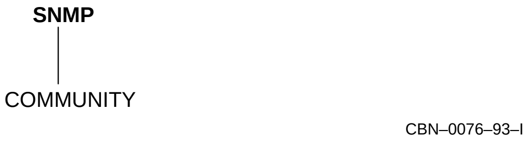
Figure A–24 SMDS Module



A.27 SNMP Module

The SNMP module implements the SNMP agent on the DECNIS.

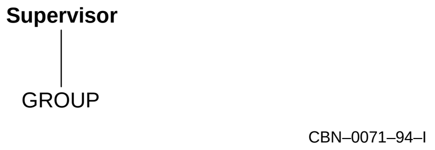
Figure A–25 SNMP Module



A.28 Supervisor Module

This controls backup groups, which associate a primary circuit and a secondary backup circuit.

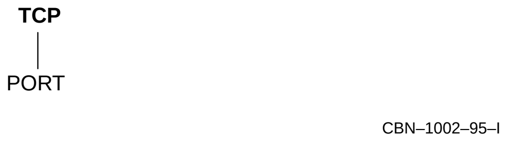
Figure A–26 Supervisor Module



A.29 TCP Module

This module manages transport layer communications with the Internet.

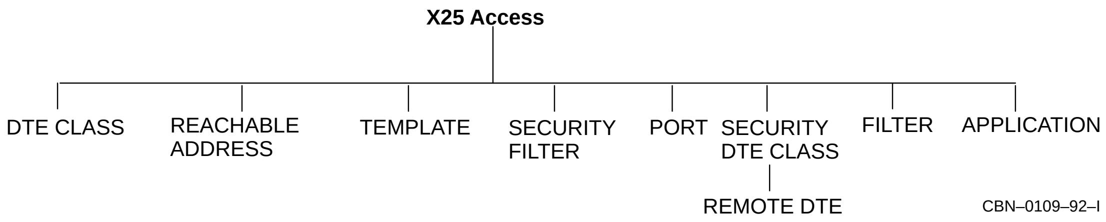
Figure A–27 TCP Module



A.30 X25 Access Module

This defines the interface between X.25 protocols and applications.

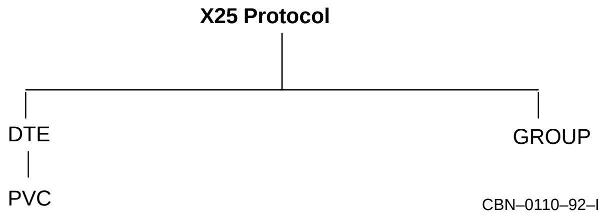
Figure A–28 X25 Access Module



A.31 X25 Protocol Module

This defines the X.25 level 3 protocol that is used to exchange packets between a DTE and a DCE. It defines the DTEs, PVCs, and CUGs recognized by an X.25 system.

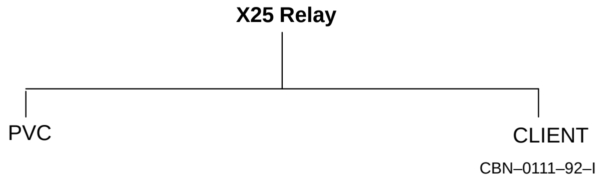
Figure A–29 X25 Protocol Module



A.32 X25 Relay Module

This defines how an X.25 call is switched from one DTE to another.

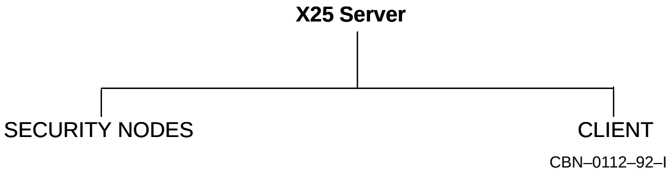
Figure A–30 X25 Relay Module



A.33 X25 Server Module

This defines how a Gateway system communicates with a Client system.

Figure A–31 X25 Server Module



B

Characteristic Attributes Supported by the DECNIS

This appendix lists the entities used with the DECNIS and provides information on their characteristic values, as implemented in the DECNIS. Note that the DECNIS implementation of certain characteristics may be more restricted than described in the generic NCL documentation. For example, the DECNIS implements some characteristics as read-only, even though these are specified as settable in the generic NCL documentation.

For detailed descriptions of the characteristics, refer to the NCL online help and/or the NCL documentation for your host operating system.

The appendix is divided alphabetically into sections by module name. Each section is further divided by entity and child entity. Each table shows the keyword, class, syntax and default value for each characteristic.

- The keyword gives the name of the attribute or characteristic.
- The classes specified are as follows:

S	Can be set at any time. Can also be read.
S/D	Can be set only when the entity is disabled.
S/C	Can be set only with the CREATE command.
R	Read-only: cannot be set.
W	Write-only: cannot be read.

- The syntax states how the characteristic is specified.

For details for the NCL commands used with each module, refer to the NCL online help and the NCL documentation for your host operating system.

B.1 ATM Connection Management Module

This section refers to the ATM Connection Management module.

B.1.1 ATM CONNECTION MANAGEMENT Characteristics

Table B–1 shows the ATM characteristics.

Table B–1 ATM CONNECTION MANAGEMENT Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version Number	V1.0.0

B.1.2 ATM CONNECTION MANAGEMENT LINE Characteristics

Table B–2 shows the ATM CONNECTION MANAGEMENT LINE characteristics.

Table B–2 ATM CONNECTION MANAGEMENT LINE Characteristics

Keyword	Class	Syntax	Default Value
ADMINISTRATION ADDRESS	S/D	Null string	–
CELL SCRAMBLING	S/D	One of: Disabled Enabled	Disabled
FLOWMASTER	S/C	One of: Disabled Enabled	Disabled
LOWER LAYER ENTITY	S/D	Local entity name	Null string
PLCP	S/D	One of: Disabled Enabled	Disabled

B.1.3 ATM CONNECTION MANAGEMENT LINE PVC Characteristics

Table B–3 shows the ATM CONNECTION MANAGEMENT LINE PVC characteristics.

Table B–3 ATM CONNECTION MANAGEMENT LINE PVC Characteristics

Keyword	Class	Syntax	Default Value
AAL TYPE	S/C	AAL5	AAL5
CIRCUIT TYPE	S/C	One of: CBR UBR ABR	CBR
ENCAPSULATION TYPE	S/C	LLC Encapsulation	LLC Encapsulation
MINIMUM GUARANTEED RATE	S	Range 1–100	1
PEAK RATE	S	Range 1–100	100
PDU SIZE	S	1–4500	4500
TRAFFIC SHAPING PRIORITY	S	Range 0–11	0
VIRTUAL CIRCUIT IDENTIFIER	S/C	Range 32–1023	–
VIRTUAL PATH IDENTIFIER	S/C	0	0

B.2 ATM Multiprotocol Encapsulation Module

This section refers to the ATM Multiprotocol Encapsulation module.

B.2.1 ATM MULTIPROTOCOL ENCAPSULATION Characteristics

Table B–4 shows the ATM MULTIPROTOCOL ENCAPSULATION characteristics.

Table B–4 ATM MULTIPROTOCOL ENCAPSULATION Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	T1.0.0

B.2.2 ATM MULTIPROTOCOL ENCAPSULATION LINK Characteristics

Table B–5 shows the ATM MULTIPROTOCOL ENCAPSULATION LINK characteristics.

Table B–5 ATM MULTIPROTOCOL ENCAPSULATION LINK Characteristics

Keyword	Class	Syntax	Default Value
LOWER LAYER ENTITY	S/D	Local entity name	Null string
MAXIMUM PDU SIZE	S/D	Range 1–4490	4490

B.3 Bridge Module

This section refers to the Bridge module.

B.3.1 BRIDGE Characteristics

Table B–6 shows the BRIDGE characteristics.

Table B–6 BRIDGE Characteristics

Keyword	Class	Syntax	Default Value
802 SPANNING TREE VERSION	R	Version number	–
BAD HELLO LIMIT	S	Range 1–255	15
BAD HELLO RESET INTERVAL	S	Range 1–255	5
BRIDGE IDENTIFIER	R	Hex string	–
FORWARD DELAY	S	Range 1–255	30
FORWARDING DATABASE NORMAL AGING TIME	S	Range 1–65535	120
FORWARDING DATABASE SHORT AGING TIME	S	Range 1–65535	30
HARDWARE VERSION	R	Version number	–
HELLO INTERVAL	S	Range 1–255	1
INTERVAL BETWEEN PORT TESTS	S	Range 1–255	60
LB100 POLL TIME	S	Range 1–65535	300
LB100 RESPONSE TIMEOUT	S	Range 1–65535	15
LB100 SPANNING TREE COMPATIBILITY	S	Auto Select or IEEE 802.1d	Auto Select
LISTEN TIME	S	Range 1–255	15
NO FRAME INTERVAL	S	Range 1–65535	300
ROOT PRIORITY	S	Range 0–65535	128
SOFTWARE VERSION	R	Version number	–
SUCCESSFUL PORT TEST THRESHOLD	S	Range 1–255	10

B.3.2 BRIDGE FILTER PID Characteristics

Table B–7 shows the BRIDGE FILTER PID characteristics.

Table B–7 BRIDGE FILTER PID Characteristics

Keyword	Class	Syntax	Default Value
PORT SET	S	Set of port names	{ }

B.3.3 BRIDGE FILTER SAP Characteristics

Table B–8 shows the BRIDGE FILTER SAP characteristics.

Table B–8 BRIDGE FILTER SAP Characteristics

Keyword	Class	Syntax	Default Value
PORT SET	S	Set of port names	{ }

B.3.4 BRIDGE FILTER TYPE Characteristics

Table B–9 shows the BRIDGE FILTER TYPE characteristics.

Table B–9 BRIDGE FILTER TYPE Characteristics

Keyword	Class	Syntax	Default Value
PORT SET	S	Set of port names	{ }

B.3.5 BRIDGE PORT Characteristics

Table B–10 shows the BRIDGE PORT characteristics.

Table B–10 BRIDGE PORT Characteristics

Keyword	Class	Syntax	Default Value
DATA LINK ENTITY	S	Local entity name	–
MANUAL DATA LINK SDU SIZE[1]	S/D	Range 1524-65535	1524
MANUAL MODE	S	TRUE/FALSE	FALSE
OTHER PID DISPOSITION	S	Forward/Filter	Forward
OTHER SAP DISPOSITION	S	Forward/Filter	Forward
OTHER TYPE DISPOSITION	S	Forward/Filter	Forward
PORT COST	S	Range 0–255	10
PORT IDENTIFIER	R	Hex string	–
PORT NUMBER	S/C	Range 1–255	–
PORT PRIORITY	S	Range 0–255	128
TYPE	S/C	LOCAL/REMOTE	LOCAL

[1] Only supported if the TYPE is REMOTE.

B.3.6 BRIDGE STATIC MULTICAST ADDRESS Characteristics

Table B–11 shows the BRIDGE STATIC MULTICAST ADDRESS characteristics.

Table B–11 BRIDGE STATIC MULTICAST ADDRESS Characteristics

Keyword	Class	Syntax	Default Value
PORT SET	S	Set of port names	{ }

B.3.7 BRIDGE STATIC PHYSICAL ADDRESS Characteristics

Table B–12 shows the BRIDGE STATIC PHYSICAL ADDRESS characteristics.

Table B–12 BRIDGE STATIC PHYSICAL ADDRESS Characteristics

Keyword	Class	Syntax	Default Value
MANUAL DESTINATION PORT	S	Port name	–
PORT SET	S	Set of port names	{ }

B.4 CHDLC Module

This section refers to the CHDLC module.

B.4.1 CHDLC Characteristic

Table B–13 shows the CHDLC characteristic.

Table B–13 CHDLC Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.4.2 CHDLC LINK Characteristics

Table B–14 shows the CHDLC LINK characteristics.

Table B–14 CHDLC LINK Characteristics

Keyword	Class	Syntax	Default Value
LOWER LAYER ENTITY ENTITY	S/D	Local entity name	–
MAXIMUM RECEIVE SDU SIZE	S/D	Range 1–65535	1500
MAXIMUM SEND SDU SIZE	S/D	Range 1–65535	1500
SLARP TRANSMISSION TIMER	S	Range 0–255	10
TYPE	S/C	Synchronous or Asynchronous	–

B.5 CSMA-CD Module

This section refers to the CSMA-CD module.

B.5.1 CSMA-CD Characteristics

Table B–15 shows the CSMA-CD characteristics.

Table B–15 CSMA-CD Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.5.2 CSMA-CD STATION Characteristics

Table B–16 shows the CSMA-CD STATION characteristics.

Table B–16 CSMA-CD STATION Characteristics

Keyword	Class	Syntax	Default Value
FAILURE DETECTION TIMER	S	Range 1–1000	20
INTERFRAME GAP	S/D	Range 1–255	26
LOWER LAYER ENTITY	S/D	Local entity name	–
STATION BUFFERS	S/D	Range 1–64	4
STATION TYPE	S/C	One of: Physical Virtual VCP Virtual PPP	–
TINYGRAM COMPRESSION	S/D	TRUE/FALSE	FALSE

B.6 DDCMP Module

This section refers to the DDCMP module.

B.6.1 DDCMP LINK Characteristics

Table B–17 shows the DDCMP LINK characteristics.

Table B–17 DDCMP LINK Characteristics

Keyword	Class	Syntax	Default Value
PHYSICAL LINE	S/D	Local entity name	–
RETRANSMIT TIMER	S	Range 1–65535	3000

B.6.2 DDCMP LOGICAL STATION Characteristics

Table B–18 shows the DDCMP LOGICAL STATION characteristics.

Table B–18 DDCMP LOGICAL STATION Characteristics

Keyword	Class	Syntax	Default Value
ADDRESS	S/D	Range 1–255	1
HOLDBACK TIMER	S	Range 0–13000	0
MAXIMUM TRANSMIT	S/D	Range 1–255	4
TRANSMIT WINDOW	S	Range 1–255	255

B.7 Device Module

This section refers to the Device module.

B.7.1 DEVICE Characteristics

Table B–19 shows the DEVICE characteristic.

Table B–19 DEVICE Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.7.2 DEVICE UNIT Characteristics

Table B–20 shows the DEVICE UNIT characteristics.

Table B–20 DEVICE UNIT Characteristics

Keyword	Class	Syntax	Default Value
AUTO LOAD	S	TRUE/FALSE	TRUE
DEVICE NAME	S/C	Simple name	–
DUMP DESTINATION	S	File name	No file name
DUMP ON ERROR	S	TRUE/FALSE	FALSE
LOAD SOURCE	S	File name	No file name

B.8 Event Dispatcher Module

This section refers to the Event Dispatcher module.

B.8.1 EVENT DISPATCHER OUTBOUND STREAM Characteristics

Table B–21 shows the EVENT DISPATCHER OUTBOUND STREAM characteristics.

Table B–21 EVENT DISPATCHER OUTBOUND STREAM Characteristics

Keyword	Class	Syntax	Default Value
TEMPLATE	S	Simple name	–
CONNECT RETRY TIMER	S	Range 0–65535	120
DISCONNECT TIMER	S	Range 0–4000 million	0
CATCHALL FILTER	S	Filter Action	Pass
CONNECT TIMER ENABLED	S	TRUE/FALSE	TRUE
SINK OBJECT	S	DNS Full Name	0:.
SINK NODE	S	DNS Full Name	0:.
SINK END USER	S	End user specification	NUMBER=82
SINK ADDRESS	S	Tower Set	{ }

B.9 FDDI Module

This section refers to the FDDI module.

B.9.1 FDDI Characteristics

Table B–22 shows the FDDI characteristics.

Table B–22 FDDI Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.9.2 FDDI STATION Characteristics

Table B–23 shows the FDDI STATION characteristics.

Table B–23 FDDI STATION Characteristics

Keyword	Class	Syntax	Default Value
COMMUNICATION PORT	S/C	Simple name	–
STATION ID	R	Octet string	–
SMT MAXIMUM VERSION ID	R	Hex string	–
SMT MINIMUM VERSION ID	R	Hex string	–
SMT VERSION ID	R	Hex string	–
TYPE	R	One of: DAC DAS NAC SAC SAS	–

B.9.3 FDDI STATION LINK Characteristics

Table B–24 shows the FDDI STATION LINK characteristics.

Table B–24 FDDI STATION LINK Characteristics

Keyword	Class	Syntax	Default Value
LINK ADDRESS	R	MAC address	–
REQUESTED TRT	S	Range 4.0—167.77208	8.0
RESTRICTED TOKEN TIMEOUT	S	Range 0—10000	1000
RING PURGER ENABLE	S	TRUE/FALSE	TRUE
VALID TRANSMISSION TIME	S	Range 2.5—5.2224	2.62144

B.9.4 FDDI STATION PHY PORT Characteristics

Table B–25 shows the FDDI STATION PHY PORT characteristics.

Table B–25 FDDI STATION PHY PORT Characteristics

Keyword	Class	Syntax	Default Value
LEM THRESHOLD	S	Range 5—8	8
PHY TYPE	S	One of: A B S M Unknown	–
PMD TYPE	S	One of: ANSI Multi-Mode ANSI Single-Mode Type 1 ANSI Single-Mode Type 2 ANSI SONET Low Power Shielded Twisted Pair ThinWire Unshielded Twisted Pair	–

B.10 FRBS Module

This section refers to the FRBS module, which implements the frame relay capabilities of the DECNIS.

B.10.1 FRBS Characteristics

Table B–26 shows the FRBS characteristic.

Table B–26 FRBS Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.10.2 FRBS CHANNEL Characteristics

Table B–27 shows the FRBS CHANNEL characteristics.

Table B–27 FRBS CHANNEL Characteristics

Keyword	Class	Syntax	Default Value
CRC TYPE	R	–	16 bit
ERROR THRESHOLD	S	Range 1–10	3
FULL ENQUIRY INTERVAL	S	Range 1–255	6
INTERFACE DESCRIPTION	S/C	Text string	–
MAXIMUM CONNECTIONS	R	–	32
MAXIMUM DATA SIZE	S	Range 1–65535	1500
MONITORED EVENTS	S	Range 1–10	4
PHYSICAL LINE	S	Local entity name	–
POLLING INTERVAL	S	Range 5–30	10
SPECIFICATION	S/C	One of: Joint ANSI CCITT	–

B.10.3 FRBS CHANNEL CONNECTION Characteristic

Table B–28 shows the FRBS CHANNEL CONNECTION characteristic.

Table B–28 FRBS CHANNEL CONNECTION Characteristic

Keyword	Class	Syntax	Default Value
PREFERRED DLCI	S/D	Range 0–4294967295	0

B.11 Hardware Module

This section refers to the hardware module.

B.11.1 HARDWARE Characteristics

Table B–29 shows the HARDWARE characteristics.

Table B–29 HARDWARE Characteristics

Keyword	Class	Syntax	Default Value
TEMPERATURE ALARM HOLDDOWN INTERVAL	S	Range 1–30	5
DEBUG FLAGS	S	Range 0–(2^{32} —1)	0
DUMP CONTROL	S	One of: NO DUMP FULL DUMP SYSTEM PROCESSOR DUMP	FULL DUMP
SELF TEST CONTROL	S	QUICK VERIFY or FULL TEST	FULL TEST

B.11.2 HARDWARE SLOT Characteristics

Table B–30 shows the HARDWARE SLOT characteristics.

Table B–30 HARDWARE SLOT Characteristics

Keyword	Class	Syntax	Default Value
FUNCTIONS DISABLED	S	One or more of: MOP load requester MOP dump requester IP load requester IP dump requester	{ }

B.12 HDLC Module

This section refers to the HDLC module.

B.12.1 HDLC Characteristic

Table B–31 shows the HDLC characteristic.

Table B–31 HDLC Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.12.2 HDLC LINK Characteristics

Table B–32 shows the HDLC LINK characteristics.

Table B–32 HDLC LINK Characteristics

Keyword	Class	Syntax	Default Value
ACKNOWLEDGE TIMER	S	Range 1–60000	3000
HOLDBACK TIMER	S/D	Range 0–60000	10
MAXIMUM UNSEQUENCED PDUS	S	Range 1–127	1
MAXIMUM DATA SIZE	S/D	Range 262–65532	1500
MINIMUM DATA SIZE	S/D	Range 262–65532	576
PHYSICAL LINE	S/D	Local Entity Name	–
PREFERRED COMPRESSION TYPES	S	One of: {DATAFLOW} {}	{}
PREFERRED CRC TYPE	S/D	16 bit, 32 bit, Either	Either
PREFERRED MAXIMUM DATA SIZE	S/D	Range 262–65532	1500
PREFERRED WINDOW SIZE	S	Range 1–127	8
PREFERRED LOCAL STATION ADDRESS	S/D	Range 2–253	2
RECEIVE BUFFERS	S/D	Range 1–128	8
RETRY MAXIMUM	S/D	Range 1–255	10

B.13 IP Services Module

This section refers to the IP Services module.

B.13.1 IP SERVICES Characteristics

Table B–33 shows the IP SERVICES characteristic.

Table B–33 IP SERVICES Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	V1.0.0

B.13.2 IP SERVICES APPLICATION Characteristics

Table B–34 shows the IP SERVICES APPLICATION characteristics. Note that these are not used.

Table B–34 IP SERVICES APPLICATION Characteristic

Keyword	Class	Syntax	Default Value
USER NAME	–	–	–
PASSWORD	–	–	–

B.13.3 IP SERVICES RESOLVER Characteristics

Table B–35 shows the IP SERVICES RESOLVER characteristics.

Table B–35 IP SERVICES RESOLVER Characteristics

Keyword	Class	Syntax	Default Value
MAXIMUM CACHE	S	0–16383	2000
REMOTE PORT	S/D	1–65535	53
LOCAL PORT	S/D	1–65535	5001

B.13.4 IP SERVICES RESOLVER NAME SERVER Characteristics

Table B–36 shows the IP SERVICES RESOLVER NAME SERVER characteristics.

Table B–36 IP SERVICES RESOLVER NAME SERVER Characteristics

Keyword	Class	Syntax	Default Value
NAME	S	Name	–
IP ADDRESS	S	IP Address	–

B.13.5 IP SERVICES RESOLVER CACHE ENTRY Characteristic

Table B–37 shows the IP SERVICES RESOLVER CACHE ENTRY characteristic.

Table B–37 IP SERVICES RESOLVER CACHE ENTRY Characteristic

Keyword	Class	Syntax	Default Value
IP ADDRESS	S	IP Address	–

B.14 LAPB Module

This section refers to the LAPB module.

B.14.1 LAPB Characteristics

Table B–38 shows the LAPB characteristic.

Table B–38 LAPB Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.14.2 LAPB LINK Characteristics

Table B–39 shows the LAPB LINK characteristics.

Table B–39 LAPB LINK Characteristics

Keyword	Class	Syntax	Default Value
ACKNOWLEDGE TIMER	S/D	Range 1–60000	See Note
HOLDBACK TIMER	S/D	Range 0–60000	See Note
INTERFACE TYPE	S/D	DTE, DCE	See Note
MAXIMUM DATA SIZE	S/D	Range 1–65532	See Note
PHYSICAL LINE	S/D	Local Entity Name	–
POLL TIMER	S/D	Range 1–600	See Note
PROFILE	S/C	Profile name	–
RECEIVE BUFFERS	S/D	Range 1–128	See Note
RETRY MAXIMUM	S/D	Range 1–255	See Note
SEQUENCE MODULUS	S/D	8, 128	See Note
WINDOW SIZE	S/D	Range 1–127	See Note

Note

The default values for the LAPB LINK characteristics are dependent on the profile for the network you are using. For details, refer to the *Network Information* manual (on line).

B.15 LLC2 Module

This section refers to the LLC2 module.

B.15.1 LLC2 Characteristic

Table B–40 shows the LLC2 characteristic.

Table B–40 LLC2 Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.15.2 LLC2 SAP Characteristic

Table B–41 shows the LLC2 SAP characteristics.

Table B–41 LLC2 SAP Characteristics

Keyword	Class	Syntax	Default Value
LAN STATION	S	Local entity name	–
LOCAL LSAP ADDRESS	S	Hex string	7E

B.15.3 LLC2 SAP LINK Characteristics

Table B–42 shows the LLC2 SAP LINK characteristics.

Table B–42 LLC2 SAP LINK Characteristics

Keyword	Class	Syntax	Default Value
ACKNOWLEDGE TIMER	S	Range 1–60000	1000
BUSY TIMER	S	Range 1–60000	1000
HOLDBACK TIMER	S	Range 0–60000	500
LOCAL RECEIVE WINDOW SIZE	S	Range 1–127	127
MAXIMUM DATA SIZE	S	Range 1–65531	1028

(continued on next page)

Table B–42 (Cont.) LLC2 SAP LINK Characteristics

Keyword	Class	Syntax	Default Value
POLL TIMER	S	Range 1–60000	3000
REJECT TIMER	S	Range 1–60000	3000
REMOTE LSAP ADDRESS	S	Hex string	7E
REMOTE MAC ADDRESS	S	LAN hardware address	00-00-00-00-00-00
RETRY MAXIMUM	S	Range 1–255	10

B.16 Loopback Application Module

This section refers to the Loopback Application module.

B.16.1 LOOPBACK APPLICATION Characteristics

Table B–43 shows the LOOPBACK APPLICATION characteristics.

Table B–43 LOOPBACK APPLICATION Characteristics

Keyword	Class	Syntax	Default Value
MAXIMUM DATA	R	Range 1500–65534	65534
MAXIMUM MIRRORS	R	Range 0–32767	0

B.17 Modem Connect Module

This section refers to the Modem Connect module.

B.17.1 MODEM CONNECT Characteristic

Table B–44 shows the MODEM CONNECT characteristic.

Table B–44 MODEM CONNECT Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.17.2 MODEM CONNECT LINE Characteristics

Table B–45 shows the MODEM CONNECT LINE characteristics.

Table B–45 MODEM CONNECT LINE Characteristics

Keyword	Class	Syntax	Default Value
ALTERNATE SPEED	S/D	Range dependent on device	0
CALL ACCEPT TIMER[1]	S	Range 0–60000	500
CARRIER LOSS TIMER[1]	S	Range 0–60000	15,000
CLOCK	S/D	EXTERNAL, INTERNAL, REFLECTED	EXTERNAL
COMMUNICATIONS MODE	S/C	Asynchronous or Synchronous	–
COMMUNICATIONS PORT	S/C	Simple name	–
CONNECTION TYPE	S/C	Nonswitched or Switched	–
DUPLEX	S/C	Full or Half	–
ENCODING	S/D	Normal or NRZI	Normal
INITIAL HOLD TIMER	S	Range 0–300	10

[1] This characteristic is controlled by the profile. The range given in this table is the range for the profile generated when you specify the NORMAL profile characteristic for the MODEM CONNECT LINE entity.

(continued on next page)

Table B–45 (Cont.) MODEM CONNECT LINE Characteristics

Keyword	Class	Syntax	Default Value
MAXIMUM CALL SETUP TIMER	S	Range 1–300	60
MAXIMUM DISABLE TRANSMIT TIMER[1]	S	Range 0–60000	500
MAXIMUM DSR DEASSERTION TIMER[1]	S	Range 0–60000	5000
MAXIMUM ENABLE TRANSMIT TIMER[1]	S	Range 1–5000	2000
MINIMUM DTR DEASSERTION TIMER[1]	S	Range 0–60000	1000
MODEM CONTROL	S/D	Full, None	Full
MODEM OPTIONS	S/D	Dialout, Direct, Rate Select	{ }
MODEM PROTOCOL FORMAT	S/D	Asynchronous, Synchronous, or HDLC	Depends on COMMUNICATIONS MODE
MODEM PROTOCOL TYPE	S/D	AT, DMCL, V25, OR V25BIS	V25BIS
PROFILE	S/C	Profile name	–
RATE SELECT	S	High, Low	High
SPEED	S/D	Range dependent on device	0
SUCCESSFUL CALL INDICATION TIMER[1]	S	Range 0–60	30
SUPPRESS TEST INDICATOR	S	TRUE/FALSE	FALSE
TRANSMIT HOLDOFF TIMER[1]	S	Range 0–60000	0

[1] This characteristic is controlled by the profile. The range given in this table is the range for the profile generated when you specify the NORMAL profile characteristic for the MODEM CONNECT LINE entity.

B.18 MOP Module

This section refers to the MOP module.

B.18.1 MOP Characteristics

Table B–46 shows the MOP characteristics.

Table B–46 MOP Characteristics

Keyword	Class	Syntax	Default Value
SUPPORTED FUNCTIONS	R	One or more of: Console Carrier Console Requester Dump Requester Dump Server Load Requester Load Server Loop Requester Query Requester Test Requester	All these functions
VERIFICATION	W	Hex string	%X0000000000000000
VERSION	R	Version number	–

B.18.2 MOP CIRCUIT Characteristics

Table B–47 shows the MOP CIRCUIT characteristics.

Table B–47 MOP CIRCUIT Characteristics

Keyword	Class	Syntax	Default Value
LINK NAME	S/D	Local entity name	–
RETRANSMIT TIMER	S	Range 1–4,000 million	4
TYPE	S/C	Circuit type	–

B.18.3 MOP CLIENT Characteristics

Table B–48 shows the MOP CLIENT characteristics.

Table B–48 MOP CLIENT Characteristics

Keyword	Class	Syntax	Default Value
ADDRESSES	S	Set of LAN addresses	{ }
CIRCUIT	S	Simple name	–
DIAGNOSTIC IMAGE	S	Sequence of file names	No file names
DUMP ADDRESS	S	Memory address	0
DUMP COUNT	S	Range 1 to (2^{32} – 1)	1
DUMP FILE	S	Sequence of file names	No file names
MANAGEMENT IMAGE	S	Sequence of file names	No file names
PHASE IV CLIENT ADDRESS	S	Phase IV address	0.0
PHASE IV CLIENT NAME	S	Phase IV name	No name
PHASE IV HOST ADDRESS	S	Phase IV address	0.0
PHASE IV HOST NAME	S	Phase IV name	No name
SCRIPT FILE	S	Sequence of file names	No file names
SECONDARY LOADER	S	Sequence of file names	No file names
SYSTEM IMAGE	S	Sequence of file names	No file names
TERTIARY LOADER	S	Sequence of file names	No file names
VERIFICATION	S	Hex string	%X0000000000000000

B.19 Multiplexed Interface Module

This section refers to the Multiplexed Interface module.

B.19.1 Multiplexed Interface Characteristics

Table B–49 shows the MULTIPLEXED INTERFACE characteristics.

Table B–49 MULTIPLEXED INTERFACE Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	T2.0.0

B.19.2 MULTIPLEXED INTERFACE LINE Characteristics

Table B–50 shows the MULTIPLEXED INTERFACE LINE characteristics.

Table B–50 MULTIPLEXED INTERFACE LINE Characteristics

Keyword	Class	Syntax	Default Value
COMMUNICATIONS PORT	S/C	Simple name	–
INTERFACE TYPE	S/C	One of: S3, E3 or OC-3	–
FRAMING TYPE	S/C	One of: DS3-CBitParity DS3-ClearChannel E3-G832 E3-G751 SONET SDH	Depends on Interface type
CODING TYPE	S/C	One of: B3ZS (for DS3) HDB3 (for E3) NRZ (for OC-3)	One of: B3ZS (for DS3) HDB3 (for E3) NRZ (for OC-3)
CHANNELIZATION MODE	S/C	Clear Channel	Clear Channel
CLOCK SOURCE	S/D	One of: Network or Local	Network
LINE LENGTH	S/D	Length in feet	0
LOOPBACK DETECTION	S/D	Disabled	Disabled

B.19.3 MULTIPLEXED INTERFACE LOGICAL CHANNEL Characteristics

Table B–51 shows the MULTIPLEXED INTERFACE LOGICAL CHANNEL characteristics.

Table B–51 MULTIPLEXED INTERFACE LOGICAL CHANNEL Characteristics

Keyword	Class	Syntax	Default Value
MAP	S/C	{0}	{0}
COMPONENT BANDWIDTH	S	One of: 34368 kbps (for E3) 44736 kbps (for T3) 155520 kbps (for OC-3)	Implementation specific
BIT INVERSION	S	FALSE	FALSE
LOOPBACK DETECTION	S	Disabled	Disabled

B.20 PPP Module

This section refers to the PPP module.

B.20.1 PPP Characteristics

Table B–52 shows the PPP entity characteristics.

Table B–52 PPP Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.20.2 PPP LINK Characteristics

Table B–53 shows the PPP LINK entity characteristics.

Table B–53 PPP LINK Characteristics

Keyword	Class	Syntax	Default Value
LOWER LAYER ENTITY	S/D	Local entity name	No entity
MAGIC NUMBER LOOP COUNT	R	Range 1—255	0
MAXIMUM CONFIGURE TRANSMISSIONS	S	Range 1—255	10
MAXIMUM CONFIGURE NAK TRANSMISSIONS	S	Range 1—255	10
MAXIMUM TERMINATE TRANSMISSIONS	S	Range	2
MINIMUM SDU SIZE	S/D	Range 1—65535	576
PREFER BRIDGE TINYGRAM COMPRESSION	S/D	TRUE/FALSE	FALSE
PREFER PROTOCOL FIELD COMPRESSION	S/D	TRUE/FALSE	FALSE
PREFER ADDRESS AND CONTROL FIELD COMPRESSION	R	TRUE/FALSE	FALSE
PREFERRED CRC SIZE	S/D	One of: 16 bits 32 bits Either	16 bits
PREFERRED MAXIMUM RECEIVE SDU SIZE	S/D	Range 1–65535	1500
PREFERRED RECEIVE CONTROL CHARACTERS TO MAP	S/D	Bitset	The set of all ASYNC characters
REQUIRED CONTROL PROTOCOLS	S	One or more of: Internet Protocol OSI Network Layer DECnet Phase IV Bridging PDU Novell IPX	All except Novell IPX

(continued on next page)

Table B–53 (Cont.) PPP LINK Characteristics

Keyword	Class	Syntax	Default Value
RETRANSMISSION TIMER	S	Range 1–255	3
TYPE	S/C	Synchronous or Asynchronous	–

B.21 Priority Module

This section refers to the Priority module.

B.21.1 PRIORITY Characteristics

Table B–54 shows the PRIORITY characteristics.

Table B–54 PRIORITY Characteristics

Keyword	Class	Syntax	Default Value
BRIDGE PID DEFAULT CLASS	S	Range 1–8	4
BRIDGE SAP DEFAULT CLASS	S	Range 1–8	4
BRIDGE TYPE DEFAULT CLASS	S	Range 1–8	4
DECNET PHASEIV DEFAULT CLASS	S	Range 1–8	4
IP PROTOCOL DEFAULT CLASS	S	Range 1–8	4
IP TCP DEFAULT CLASS	S	Range 1–8	4
IP TOS DEFAULT CLASS	S	Range 1–8 or Ignore TOS Field	Ignore TOS Field
IP UDP DEFAULT CLASS	S	Range 1–8	4
IPX DEFAULT CLASS	S	Range 1–8	4
OSI DEFAULT CLASS	S	Range 1–8	4
VERSION	R	Version number	–

B.21.2 PRIORITY GROUP Characteristics

Table B–55 shows the PRIORITY GROUP characteristics.

Table B–55 PRIORITY GROUP Characteristics

Keyword	Class	Syntax	Default Value
ASSIGNED CLASS	S	Range 1–8	1

B.21.3 PRIORITY FILTER Characteristics

Table B–56 shows the PRIORITY FILTER characteristics.

Table B–56 PRIORITY FILTER Characteristics

Keyword	Class	Syntax	Default Value
DESTINATION ADDRESS MASK	S	n.n.n.n	0.0.0.0
DESTINATION ADDRESS VALUE	S	IP address	0.0.0.0
DESTINATION PORT MASK	S		0
DESTINATION PORT VALUE	S	0–65535	0
ICMP MESSAGE MASK	S	0–255	0
ICMP MESSAGE TYPE	S	0–255	0
INBOUND INTERFACE	S	PRIORITY INTERFACE *interface-name*	–
IP PROTOCOL MASK	S	Range 0–255	255
IP PROTOCOL VALUE	S	Range 0–255	0
SOURCE ADDRESS MASK	S	n.n.n.n	0.0.0.0
SOURCE ADDRESS VALUE	S	IP address	0.0.0.0
SOURCE PORT MASK	S	Range 0–65535	0
SOURCE PORT VALUE	S	Range 0–65535	0
TRACE	S	True or False	False

B.21.4 PRIORITY INTERFACE Characteristics

Table B–57 shows the PRIORITY INTERFACE characteristics.

Table B–57 PRIORITY INTERFACE Characteristics

Keyword	Class	Syntax	Default Value
CLASS 1 ASSIGNED QUEUE	S	Range 1–8	1
CLASS 2 ASSIGNED QUEUE	S	Range 1–8	2
CLASS 3 ASSIGNED QUEUE	S	Range 1–8	3
CLASS 4 ASSIGNED QUEUE	S	Range 1–8	4
CLASS 5 ASSIGNED QUEUE	S	Range 1–8	5
CLASS 6 ASSIGNED QUEUE	S	Range 1–8	6
CLASS 7 ASSIGNED QUEUE	S	Range 1–8	7
CLASS 8 ASSIGNED QUEUE	S	Range 1–8	8

B.21.5 PRIORITY INTERFACE FILTER Characteristics

Table B–58 shows the PRIORITY INTERFACE FILTER characteristics.

Table B–58 PRIORITY INTERFACE FILTER Characteristics

Keyword	Class	Syntax	Default Value
DESTINATION ADDRESS MASK	S	n.n.n.n	0.0.0.0
DESTINATION ADDRESS VALUE	S	IP address	0.0.0.0
DESTINATION PORT MASK	S	0–65535	0
DESTINATION PORT VALUE	S	0–65535	0
ICMP MESSAGE MASK	S	0–255	0
ICMP MESSAGE TYPE	S	0–255	0
INBOUND INTERFACE	S	PRIORITY INTERFACE *interface-name*	–
IP PROTOCOL MASK	S	0–255	0

(continued on next page)

Table B–58 (Cont.) PRIORITY INTERFACE FILTER Characteristics

Keyword	Class	Syntax	Default Value
IP PROTOCOL VALUE	S	0–255	0
SOURCE ADDRESS MASK	S	n.n.n.n	0.0.0.0
SOURCE ADDRESS VALUE	S	IP address	0.0.0.0
SOURCE PORT MASK	S	0–65535	0
SOURCE PORT VALUE	S	0–65535	0
TRACE	S	True or False	False

B.21.6 PRIORITY INTERFACE QUEUE Characteristics

Table B–59 shows the PRIORITY INTERFACE QUEUE characteristics.

Table B–59 PRIORITY INTERFACE QUEUE Characteristics

Keyword	Class	Syntax	Default Value
ABSOLUTE PRIORITY	S	Range 0–255	1
MAXIMUM QUEUE LENGTH	S	Range 0–65535	Queue 1 = 65535 Queue 2 to 8 = 100
WEIGHTING	S	Range 0–255	1

B.21.7 PRIORITY PACKET Characteristics

Table B–60 shows the PRIORITY PACKET characteristics.

Table B–60 PRIORITY PACKET Characteristics

Keyword	Class	Syntax	Default Value
BRIDGE ETHERNET PROTOCOL TYPE	S/D	Protocol type in the form *nn-nn*	05-DD
BRIDGE PID VALUE	S/D	Protocol type in the form *nn-nn-nn-nn-nn*	00-00-00-00-00
BRIDGE SAP VALUE	S/D	Protocol type in the form *nn*	00

(continued on next page)

Table B–60 (Cont.) PRIORITY PACKET Characteristics

Keyword	Class	Syntax	Default Value
IP PORT RANGE	S/D	Range from [0..65535]	[0..0]
IP PROTOCOL	S/D	Range 0–65535	0
IP TOS VALUE	S/D	{PRECEDENCE=P,SERVICE=S} Where 'P' is one of: network control internetwork control critic flash override immediate priority routine ignore and 'S' is one of: minimize delay maximize throughput maximize reliability minimize monetary cost normal service ignore	{PRECEDENCE =routine, SERVICE =minimize monetary cost}
MATCH ACTION	S	Name of a PRIORITY GROUP or PRIORITY PATTERN	–
PACKET LENGTH	S/D	Range from [0..65535]	[0..0]
TYPE	S/D	One of: BRIDGE PID BRIDGE SAP BRIDGE TYPE DECNET IP PROTOCOL IP TCP IP TOS IP UDP	–

B.21.8 PRIORITY PATTERN Characteristics

Table B–61 shows the PRIORITY PATTERN characteristics.

Table B–61 PRIORITY PATTERN Characteristics

Keyword	Class	Syntax	Default Value
FROM	S/D	One of: Start Next ISO Layer	Start
MASK	S/D	Hex string	–
MATCH ACTION	S/D	Name of PRIORITY GROUP or PRIORITY PATTERN	–
MISMATCH ACTION	S/D	Name of PRIORITY GROUP or PRIORITY PATTERN	–
OFFSET	S/D	Range 0–62	0
STRING	S/D	Hex string	–
TRACE MATCH	S/D	True or False	False
TRACE MISMATCH	S/D	True or False	False

B.22 Routing Module

This section refers to the Routing module.

B.22.1 ROUTING Characteristics

Table B–62 shows the ROUTING entity characteristics.

Table B–62 ROUTING Characteristics

Keyword	Class	Syntax	Default Value
AUTHENTICATION TYPES SUPPORTED	R	One or more of: Circuit Area Domain	–
AUTONOMOUS SYSTEM NUMBER	S/D	Range 0–65535	0
DEFAULT ESHELLO TIMER	S/D	Range 1–65535	600
DR ISISHELLO TIMER	S	Range 1–65535	1
GENERATE CHECKSUMS	S	TRUE/FALSE	FALSE
HOLDING MULTIPLIER	S	Range 2–63	3
INTEGRATED ISIS PROTOCOLS	S/D	One or more of: ISO8473 IP	Value of PROTOCOLS attribute
IP L1 CONNECTIVITY	S/D	Min: 10	20
IP L2 CONNECTIVITY	S/D	Min: 10	20
IP REASSEMBLY TIME	S	Range 1–255	10
ISIS HOLDING MULTIPLIER	S/D	Range 2–63	10
L1 CONNECTIVITY	S/D	Min: 10	20
L2 CONNECTIVITY	S/D	Min: 10	20
LIFETIME	S	Range 2–255	63
MANUAL AREA ADDRESSES	S	Set of Area Address	{ }
MANUAL IP ADDRESS	S	IP address	{0.0.0.0}
MANUAL L1 ALGORITHM	S/D	Routing Vector, Link State	Routing Vector

(continued on next page)

Table B–62 (Cont.) ROUTING Characteristics

Keyword	Class	Syntax	Default Value
MANUAL L2 ALGORITHM	S/D	Routing Vector, Link State	Routing Vector
MAXIMUM AGE	S/D	Range 1–65535	1200
MAXIMUM AREA ADDRESSES	R	–	–
MAXIMUM BUFFERS	R	Range 1–200	100
MAXIMUM DA ADJACENCIES	S/D	Min: 0	160
MAXIMUM ENDSYSTEM ADJACENCIES	S/D	Min: 10	5120
MAXIMUM IP EXTERNAL ADJACENCIES	S/D	Min: 10	700
MAXIMUM IP EXTERNAL DESTINATIONS	S/D	Min: 100	500
MAXIMUM IP L1 DESTINATIONS	S/D	Min: 100	250
MAXIMUM IP L2 DESTINATIONS	S/D	Min: 100	890
MAXIMUM IP LOCAL ADJACENCIES	S/D	Min: 1	50
MAXIMUM IP REACHABLE DESTINATIONS	S/D	Min: 0	200
MAXIMUM L1 LSPS	S/D	Min: 10	400
MAXIMUM L1 ROUTERS	S/D	Min: 10	100
MAXIMUM L1 ROUTING DESTINATIONS	S/D	Min: 100	5280
MAXIMUM L2 LSPS	S/D	Min: 10	1177
MAXIMUM L2 ROUTERS	S/D	Min: 10	512
MAXIMUM L2 ROUTING DESTINATIONS	S/D	Min: 100	456
MAXIMUM LSP GENERATION INTERVAL	S/D	Range 60–900	900
MAXIMUM MANUAL ADJACENCIES	S/D	Min: 0	60
MAXIMUM PATH SPLITS	S/D	Range 1–4	1

(continued on next page)

Table B–62 (Cont.) ROUTING Characteristics

Keyword	Class	Syntax	Default Value
MAXIMUM REACHABLE DESTINATIONS	S/D	Min: 0	200
MAXIMUM ROUTER ADJACENCIES	S/D	Min: 10	170
MINIMUM LSP GENERATION INTERVAL	S/D	Range 1–65535	30
PHASEIV ADDRESS	S/D	Phase IV Address	0.0
PHASEIV AREA MAXIMUM COST	S	Range 1–1022	1022
PHASEIV AREA MAXIMUM HOPS	S	Range 1–30	30
PHASEIV BROADCAST ROUTING TIMER	S	Range 1–65535	10
PHASEIV MAXIMUM ADDRESS	S/D	Range 1–1023	1023
PHASEIV MAXIMUM AREA	S/D	Range 1–63	63
PHASEIV MAXIMUM COST	S	Range 1–1022	1022
PHASEIV MAXIMUM HOPS	S	Range 1–30	30
PHASEIV MAXIMUM VISITS	S	Range 1–63	63
PHASEIV PREFIX	S/D	Address Prefix	49
POLL ESHELLO TIMER	S/D	Range 1–65535	50
PROTOCOLS	S/C	One or more of: IP ISO8473	–
REDIRECT HOLDING TIME	S	Range 1–65535	600
SEGMENT BUFFER SIZE	S	Range 0–65535	576
SEND SOURCE QUENCH	S	TRUE/FALSE	FALSE
SOURCE QUENCH INTERVAL	S	Range 1–65535	100
SUMMARY ADDRESSES	S/D	Set of Summary Addresses	None
TIME TO LIVE	S	Range 1–255	35

(continued on next page)

Table B–62 (Cont.) ROUTING Characteristics

Keyword	Class	Syntax	Default Value
TYPE	S/C	One of: L1ROUTER L2ROUTER	–
VERSION	R	Version number	–

B.22.2 ROUTING CIRCUIT Characteristics

Table B–63 shows the ROUTING CIRCUIT characteristics.

Table B–63 ROUTING CIRCUIT Characteristics

Keyword	Class	Syntax	Default Value
ALTERNATIVE SUBNET ADDRESSES	S/D	Set of subnet addresses	{ }
ARP HOLDING TIME	S	Range 60–65535	600
ARP RESPONSE WAITING TIME	S	Range 1–5	3
BOOTP RELAY	S	IP address	None
BOOTP SERVERS	S	Set of IP addresses	None
DATA LINK ENTITY	S/D	Local Entity Name	–
DNA NEIGHBOR	S/D	TRUE/FALSE	TRUE
ENABLE PHASEIV ADDRESS	S/D	TRUE/FALSE	TRUE
ENCAPSULATION DESTINATION IP ADDRESS	S/D	IP address	0.0.0.0
EXPLICIT RECEIVE VERIFICATION	S	TRUE/FALSE	TRUE
HELLO TIMER	S	Range 1–32,767	10
IDLE TIMER	S/D	Range 1–65535	30
INITIAL MINIMUM TIMER	S/D	Range 1–65535	55
IP TEMPLATE	S/D	Simple Name	""
ISIS HELLO TIMER	S	Range 1–32,767	3

(continued on next page)

Table B–63 (Cont.) ROUTING CIRCUIT Characteristics

Keyword	Class	Syntax	Default Value
L1 COST	S	Range 1–63	18
L1 ROUTER PRIORITY	S	Range 1–127	64
L2 COST	S	Range 1–63	18
L2 ROUTER PRIORITY	S	Range 1–127	64
MANUAL DATA LINK SDU SIZE	S/D	Range[1] 128–65535	1492
MANUAL IP DATA LINK SDU SIZE	S/D	Range[1] 128–65535	1492
MANUAL L2ONLY MODE	S/D	TRUE/FALSE	FALSE
MAXIMUM ARP RETRIES	S	Range 1–5	3
MAXIMUM CALL ATTEMPTS	S	Range 0–255	10
MAXIMUM ENDSYSTEM ADJACENCIES	S/D	Min: 0	2480
MAXIMUM SVC ADJACENCIES	S/D	Range 1–65535	1
NEIGHBOR IP ADDRESS	S	IP address	{0.0.0.0}
NETWORK PROTOCOLS	S/D	Set of protocols	Protocols specified in ROUTING PROTOCOLS
ORIGINATING QUEUE LIMIT	S/D	Range 1–4000 million	2
PROXY ARP	S	On or Off	Off
PHASE IV ROUTING VECTOR TIMER	S	Range 1–65535	30
RECALL TIMER	S/D	Range 0–65535	60
RECEIVE VERIFIER	W[2]	Hex string	Null string
SEGMENTED RV BLOCK SIZE	S	Range 0–65535	0
SEGMENTED RV TIMER	S	Range 1–65535	5

[1] If the DECNIS is connected to a DECnet/OSI router which is running as a link state router, this characteristic must be set to at least 1492.

[2] You can set this characteristic at any time. However, the new setting does not take effect until the circuit is reenabled.

(continued on next page)

Table B–63 (Cont.) ROUTING CIRCUIT Characteristics

Keyword	Class	Syntax	Default Value
SUBNET ADDRESS	S/D	Subnet Address	{0.0.0.0, 0.0.0.0}
TEMPLATE	S/D	Simple Name	No name
TRANSLATEIVTOV	S/D	IF POSSIBLE/IF REQUIRED	IF REQUIRED
TRANSMIT VERIFIER	W[2]	Octet String (0–38 characters)	Null string
TYPE	S/C	One of: ATM PERMANENT CSMA-CD DDCMP FDDI HDLC PPP SMDS VIRTUAL BROADCAST VIRTUAL POINT TO POINT X25DA X25 PERMANENT X25 STATIC INCOMING X25 STATIC OUTGOING	–

X25 FILTERS	S/D	Set of Simple Names	If TYPE is X25 STATIC INCOMING, {Phasevrouting, Phaseivrouting}
			If TYPE is X25 DA, {Phasevrouting}

[2] You can set this characteristic at any time. However, the new setting does not take effect until the circuit is reenabled.

B.22.3 ROUTING CIRCUIT IGMP Characteristics

Table B–64 shows the ROUTING CIRCUIT IGMP characteristics.

Table B–64 ROUTING CIRCUIT IGMP Characteristics

Keyword	Class	Syntax	Default Value
GROUPS	S/D	For FDDI circuits only. Set of Class D IP destination addresses Range 224.0.1.0 to 239.255.255.255	{ }
HOLDING MULTIPLIER	S/D	Range 1 to 65535	3
QUERY INTERVAL	S/D	Range 1 to 65535	10

B.22.4 ROUTING CIRCUIT IP ADDRESS TRANSLATION Characteristics

Table B–65 shows the ROUTING CIRCUIT IP ADDRESS TRANSLATION characteristic.

Table B–65 ROUTING CIRCUIT IP ADDRESS TRANSLATION Characteristics

Keyword	Class	Syntax	Default Value
LAN ADDRESS	S/C	LAN address	–

B.22.5 ROUTING CIRCUIT IP REACHABLE ADDRESS Characteristics

Table B–66 shows the ROUTING IP REACHABLE ADDRESS characteristics.

Table B–66 ROUTING CIRCUIT IP REACHABLE ADDRESS Characteristics

Keyword	Class	Syntax	Default Value
DESTINATION	S/C	IP address	–
DTE ADDRESSES	S/D	Set of DTE addresses	{ }
METRIC	S/D	Range 1–Maximum link cost	20
NEXT HOP	S/D	IP Address	0.0.0.0
PREFERENCE	S	Range 0–255	10

B.22.6 ROUTING CIRCUIT IP STANDBY Characteristics

Table B–67 shows the ROUTING CIRCUIT IP STANDBY characteristics.

Table B–67 ROUTING CIRCUIT IP STANDBY Characteristics

Keyword	Class	Syntax	Default Value
MODE	S/C	MAC	MAC
ROUTERS	S/D	Set of MAC selectors and associated IP addresses	{ }
MONITORED CIRCUITS	S	Set of routing circuits	{ }
PRIORITY	S	Range 1–127	64
HOLDING MULTIPLIER	S	Range 2–63	3

B.22.7 ROUTING CIRCUIT NETWORK PROTOCOL Characteristics

Table B–68 shows the ROUTING CIRCUIT NETWORK PROTOCOL characteristics.

Table B–68 ROUTING CIRCUIT NETWORK PROTOCOL Characteristics

Keyword	Class	Syntax	Default Value
NETWARE NETWORK NUMBER	S/D	Hex string	0
NETWARE DATA LINK ENCAPSULATION	S/D	One of: Ethernet; 802.2; SNAP; Novell	Ethernet
NETWARE NETBIOS BROADCAST	S/D	ON or OFF	OFF
PERIODIC ROUTING PROTOCOL TIMER	S/D	Range 0–64000	30
APPLETALK MANUAL NETWORK RANGE	S/D	Range of values: lower and upper limits must be between 0 and 65279	0..0
APPLETALK MANUAL NETWORK ADDRESS	S/D	AppleTalk address	0.0
APPLETALK PRIMARY ZONE	S/D	Simple name	No name
APPLETALK SECONDARY ZONES	S/D	Set of simple names	{ }

B.22.8 ROUTING CIRCUIT NETWORK PROTOCOL ADJACENCY Characteristics

Table B–69 shows the ROUTING CIRCUIT NETWORK PROTOCOL ADJACENCY characteristic.

Table B–69 ROUTING CIRCUIT NETWORK PROTOCOL ADJACENCY Characteristics

Keyword	Class	Syntax	Default Value
ENCAPSULATION IP ADDRESS	S/D	IP address	0.0.0.0

B.22.9 ROUTING CIRCUIT REACHABLE ADDRESS Characteristics

Table B–70 shows the ROUTING CIRCUIT REACHABLE ADDRESS characteristics.

Table B–70 ROUTING CIRCUIT REACHABLE ADDRESS Characteristics

Keyword	Class	Syntax	Default Value
ADDRESS PREFIX	S/C	NSAP address	No address
ADJACENCY DEPENDENT	S/D	TRUE/FALSE	FALSE
COST	S/D	Range 1–1023	18
DATA FORMAT	S/D	PHASEV/PHASEIV	PHASE V
DTE ADDRESSES	S/D	Set of DTE Addresses	{ }
LAN ADDRESS	S/D	ID	00-00-00-00-00-00
MAPPING	S/D	MANUAL/X.121	X.121
METRIC TYPE	S	INTERNAL/EXTERNAL	INTERNAL
TYPE	S/D	OUTBOUND/INBOUND	OUTBOUND

B.22.10 ROUTING CIRCUIT ROUTER DISCOVERY Characteristics

Table B–71 shows the ROUTING CIRCUIT ROUTER DISCOVERY characteristics.

Table B–71 ROUTING CIRCUIT ROUTER DISCOVERY Characteristics

Keyword	Class	Syntax	Default Value
ADDRESSES	S	Set of IP addresses with preferences	{ }
ADVERTISEMENT ADDRESS	S	255.255.255.255 or 224.0.0.1	224.0.0.1
MAX ADVERTISEMENT INTERVAL	S	Range 4–65535	600
MIN ADVERTISEMENT INTERVAL	S	Range: 3 to MAX ADVERTISEMENT INTERVAL	0.75 * MAX ADVERTISEMENT INTERVAL

(continued on next page)

Table B–71 (Cont.) ROUTING CIRCUIT ROUTER DISCOVERY Characteristics

Keyword	Class	Syntax	Default Value
LIFETIME	S	Range: MAX ADVERTISEMENT INTERVAL–9000	3 * MAX ADVERTISE-MENT INTERVAL

B.22.11 ROUTING CIRCUIT UDP SERVER Characteristics

Table B–72 shows the ROUTING CIRCUIT UDP SERVER characteristics.

Table B–72 ROUTING CIRCUIT UDP SERVER Characteristics

Keyword	Class	Syntax	Default Value
ADDRESS	S/C	IP address	–
STATE	R	On or Off	Off
PORTS	S/D	Set of Port numbers. Range 0–65535	{ }

B.22.12 ROUTING CONTROL PROTOCOL Characteristics

Table B–73 shows the ROUTING CONTROL PROTOCOL characteristics.

Table B–73 ROUTING CONTROL PROTOCOL Characteristics

Keyword	Class	Syntax	Default Value
BGP AGGREGATION	S/D	Set of local entity names	{ }
IP SINKS	S/D	Set of IP addresses	{ }
IP SOURCES	S/D	Set of IP addresses	{ }
IP SUBNETS	S/D	Set of IP subnet addresses	{ }
MAXIMUM ADJACENCIES	S/D	Range 0–65535	25
MAXIMUM DESTINATIONS	S/D	Range 0–65535	300
OSPF AREA ID	S/D	IP address	0.0.0.0
OSPF AREA RANGE	S/D	Set of IP subnet addresses	{ }

(continued on next page)

Table B–73 (Cont.) ROUTING CONTROL PROTOCOL Characteristics

Keyword	Class	Syntax	Default Value
OSPF AUTHENTICATION	S/D	One of: SIMPLE PASSWORD NONE	NONE
OSPF AUTONOMOUS SYSTEM BOUNDARY ROUTER	S/D	TRUE/FALSE	FALSE
OSPF AVERAGE AREA NETWORKS	S/D	Range 1–65535	40
OSPF AVERAGE CONNECTED ROUTERS	S/D	Range 1–255	10
OSPF AVERAGE EXTERNAL CONNECTIVITY	S/D	Range 0–255	2
OSPF GENERAL INSTANCE	S/D	OSPF GENERAL entity name	OSPF-General
OSPF MAXIMUM AREA INTERFACES	S/D	Range 1–255	3
OSPF MAXIMUM BOUNDARY ROUTERS	S/D	Range 0–255	4
OSPF MAXIMUM CONNECTED AREAS	S/D	Range 1–255	2
OSPF MAXIMUM EXTERNAL ROUTES	S/D	Range 0–65535	50
OSPF MAXIMUM NETWORK ROUTERS	S/D	Range 1–255	5
OSPF MAXIMUM SYSTEM NETWORKS	S	Range 1–255	25
OSPF STUB AREA	S/D	TRUE/FALSE	FALSE
OSPF STUB DEFAULT METRIC	S	Range 1–16777215	1
OSPF VERSION	R	Version number	–
POISONED REVERSE	S/D	TRUE/FALSE	FALSE
PREFERENCE	S/D	Range 0–255	Depends on protocol

(continued on next page)

Table B–73 (Cont.) ROUTING CONTROL PROTOCOL Characteristics

Keyword	Class	Syntax	Default Value
RIP MODE	S/D	BROADCAST/POINT-TO-POINT	BROADCAST
RIP STATE	S/D	One of: OFF Receive Only Send and Receive Send Only	Send and Receive
TYPE	S/C	One of: BGP EGP ISIS Level 1 ISIS Level 2 IP RIP Local OSPF General OSPF Area Static	–

B.22.13 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT Characteristics

Table B–74 shows the ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT characteristics.

Table B–74 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT Characteristics

Keyword	Class	Syntax	Default Value
CIRCUIT	S/D	Local entity name	" "
METRIC	S	Range 0–4294967295	Depends on TYPE
OSPF AUTHENTICATION KEY	W	An octet string	0
OSPF DEAD INTERVAL	S	1–2147483647	40
OSPF HELLO INTERVAL	S	1–65535	10
OSPF POLL INTERVAL	S	1–2147483647	120
OSPF PRIORITY	S	Range 0–255	1
OSPF RETRANSMISSION INTERVAL	S	1–3600	5
OSPF TRANSIT AREA	S/D	Area name	" "
OSPF TRANSIT DELAY	S	1–3600	1
OSPF VIRTUAL NEIGHBOR ROUTER ID	S/D	IP address	0.0.0.0
TYPE	S/D	One of: BROADCAST POINT TO POINT NBMA VIRTUAL	–

B.22.14 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT ADJACENCY Characteristics

Table B–75 shows the ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT ADJACENCY characteristics.

Table B–75 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT ADJACENCY Characteristics

Keyword	Class	Syntax	Default Value
OSPF PRIORITY	S	Range 1–255	1
OSPF NEIGHBOR ADDRESS	S	IP address	0.0.0.0

B.22.15 ROUTING CONTROL PROTOCOL NEIGHBOR Characteristics

Table B–76 shows the ROUTING CONTROL PROTOCOL NEIGHBOR characteristics.

Table B–76 ROUTING CONTROL PROTOCOL NEIGHBOR Characteristics

Keyword	Class	Syntax	Default Value
AUTONOMOUS SYSTEM	S/D	Autonomous system number	0
CONNECT TIME	S/D	Range 1–65535	120
HOLD TIME	S/D	Range 1–65535	90
INITIAL TIME	S/D	Range 1–65535	12
IP ADDRESS	S/D	IP address	0.0.0.0
KEEPALIVE TIME	S/D	Range 1–65535	30

B.22.16 ROUTING CONTROL PROTOCOL PROPAGATE DESTINATION Characteristics

Table B–77 shows the ROUTING CONTROL PROTOCOL PROPAGATE DESTINATION characteristics.

Table B–77 ROUTING CONTROL PROTOCOL PROPAGATE DESTINATION Characteristics

Keyword	Class	Syntax	Default Value
ADDRESS MATCH TYPE	S/D	Exact or Prefix	Prefix
ANNOUNCE METRIC	S/D	Range 1–255	PROTOCOL entity
ANNOUNCE NEIGHBORS	S/D	Set of autonomous system numbers	{ }
ANNOUNCE NEXT HOP	S/D	IP address	0.0.0.0
ANNOUNCE ROUTE ATTRIBUTES	S/D	One or more of: ISIS Internal ISIS External OSPF External Type 1 OSPF External Type 2 OSPF Intra Area OSPF Inter Area	{ }
DISCRIMINATOR TYPE	S/D	One of: None Fixed Passthrough	None
DOMAINS	S/D	Set of local entity names	{ }
EXCLUDED AUTONOMOUS SYSTEMS	S/D	Set of autonomous system numbers	{ }
FILTER ACTION	S/C	One of: Pass Block	Block
MULTI EXIT DISCRIMINATOR	S/D	Range 0–65535	0
NEIGHBOR AUTONOMOUS SYSTEM	S/D	Autonomous system number	0

(continued on next page)

Table B–77 (Cont.) ROUTING CONTROL PROTOCOL PROPAGATE DESTINATION Characteristics

Keyword	Class	Syntax	Default Value
ORIGIN	S/D	One of: IGP EGP Incomplete Any	Any
ORIGINATING AUTONOMOUS SYSTEM	S/D	Autonomous system number	0 (ignored)
SOURCE ROUTE ATTRIBUTES	S/C	One or more of: ISIS Internal ISIS External OSPF External Type 1 OSPF External Type 2 OSPF Intra Area OSPF Inter Area	{ }
SOURCES	S/D	Set of local entity names	{ }
TAG	S/D	Range 1–65535 or 0 (any tag value acceptable)	0

B.22.17 ROUTING CONTROL PROTOCOL RECEIVE DESTINATION Characteristics

Table B–78 shows the ROUTING CONTROL PROTOCOL RECEIVE DESTINATION characteristics.

Table B–78 ROUTING CONTROL PROTOCOL RECEIVE DESTINATION Characteristics

Keyword	Class	Syntax	Default Value
ADDRESS MATCH TYPE	S/D	Exact or Prefix	Prefix
DOMAINS	S/D	Set of local entity names	{ }
EXCLUDED AUTONOMOUS SYSTEMS	S/D	Set of autonomous system numbers	{ }
FILTER ACTION	S/C	Pass or Block	Block

(continued on next page)

Table B–78 (Cont.) ROUTING CONTROL PROTOCOL RECEIVE DESTINATION Characteristics

Keyword	Class	Syntax	Default Value
NEIGHBOR AUTONOMOUS SYSTEM	S/D	Autonomous system number	0
ORIGIN	S/D	One of: IGP BGP Incomplete Any	Any
ORIGINATING AUTONOMOUS SYSTEM	S/D	Autonomous system number	0
PREFERENCE	S/D	Range 0–255	Preference in parent Routing Control Protocol
SOURCE ROUTE ATTRIBUTES	S/C	One or more of: ISIS Internal ISIS External OSPF External Type 1 OSPF External Type 2 OSPF Intra Area OSPF Inter Area	–

B.22.18 ROUTING EGP GROUP Characteristics

Table B–79 shows the ROUTING EGP GROUP characteristics.

Table B–79 ROUTING EGP GROUP Characteristics

Keyword	Class	Syntax	Default Value
AUTONOMOUS SYSTEM NUMBER	S/D	Range 0–65535	0
MAXIMUM ACTIVE NEIGHBORS	S/D	Range 0–255	1
SEND METRIC CLASSES	S/D	Set of {INTERNAL, EXTERNAL}	{INTERNAL}

B.22.19 ROUTING EGP GROUP EGP NEIGHBOR Characteristics

Table B–80 shows the ROUTING EGP GROUP EGP NEIGHBOR characteristics.

Table B–80 ROUTING EGP GROUP EGP NEIGHBOR Characteristics

Keyword	Class	Syntax	Default Value
CIRCUIT	S/C	Simple name	No circuit
IP ADDRESS	S/C	IP address	No address
SOURCE NETWORK	S	IP address	{0.0.0.0}

B.22.20 ROUTING NETWORK PROTOCOL DOMAIN Characteristics

Table B–81 shows the ROUTING NETWORK PROTOCOL DOMAIN characteristics.

Table B–81 ROUTING NETWORK PROTOCOL DOMAIN Characteristics

Keyword	Class	Syntax	Default Value
IP ADDRESSES	S/D	Set of subnetwork addresses	{ }

B.22.21 ROUTING PERMITTED NEIGHBOR Characteristics

Table B–82 shows the ROUTING PERMITTED NEIGHBOR characteristics.

Table B–82 ROUTING PERMITTED NEIGHBOR Characteristics

Keyword	Class	Syntax	Default Value
ID	S/C	Node-ID	–
VERIFIER	W[1]	Octet String	No verifier

[1] You can set the VERIFIER at any time. However, a new setting does not take effect until the circuit is reenabled.

B.23 Session Control Module

This section refers to the Session Control module.

B.23.1 SESSION CONTROL Characteristics

Table B–83 shows the SESSION CONTROL characteristics.

Table B–83 SESSION CONTROL Characteristics

Keyword	Class	Syntax	Default Value
ADDRESS UPDATE INTERVAL	S	Range 0–65535	10
INCOMING PROXY	S	TRUE/FALSE	TRUE
INCOMING TIMER	S	Range 0–65535	45
MAINTAIN BACKWARD SOFT LINKS	S	TRUE/FALSE	FALSE
MAXIMUM KNOWN TOWERS	S	Range 0–65535	0
MODIFY ACS	S	TRUE/FALSE	TRUE
NODE SYNONYM DIRECTORY	S	Full name	DNA$NODESYNONYM
NON PRIVILEGED PASSWORD	W	Simple Name	–
NON PRIVILEGED USER	W	Simple Name	–
PRIVILEGED PASSWORD	W	Simple Name	–
PRIVILEGED USER	W	Simple Name	–
OUTGOING PROXY	S	TRUE/FALSE	TRUE
OUTGOING TIMER	S	Range 0–65535	60
SOFT LINK TIMER	S	Range 0–65535	30
UPDATE RETRY INTERVAL	S	Range 0–65535	60
VERSION	R	Version number	–

B.23.2 SESSION CONTROL KNOWN TOWER Characteristics

Table B–84 shows the SESSION CONTROL KNOWN TOWER characteristics.

Table B–84 SESSION CONTROL KNOWN TOWER Characteristics

Keyword	Class	Syntax	Default Value
TOWERS	S	Tower Set	{ }

B.24 SMDS Module

This section refers to the SMDS module.

B.24.1 SMDS Characteristics

Table B–85 shows the SMDS characteristics.

Table B–85 SMDS Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.24.2 SMDS STATION Characteristics

Table B–86 shows the SMDS STATION characteristics.

Table B–86 SMDS STATION Characteristics

Keyword	Class	Syntax	Default Value
LOCAL ADDRESS	S/D	MAC address	–
GROUP ADDRESS	S/D	MAC address	–
HEARTBEAT TIMER	S/D	Range 0–255	0
HEARTBEAT THRESHOLD	S/D	Range 0–255	0
LOWER LAYER ENTITY	S/D	Local entity name	–
CRC TYPE	S/D	16-BIT/32-BIT	32-BIT
INTERFRAME GAP	S/D	Range 0–255	0

B.25 SNMP Module

This section refers to the SNMP module.

B.25.1 SNMP Characteristics

Table B–87 shows the SNMP characteristics.

Table B–87 SNMP Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.25.2 SNMP COMMUNITY Characteristics

Table B–88 shows the SNMP COMMUNITY characteristics.

Table B–88 SNMP COMMUNITY Characteristics

Keyword	Class	Syntax	Default Value
PRIVATE NAME	S	Community name suffix	–
SNMP TYPE	S/C	One or more of: Agent Director	{Agent}
ACCESS	S	Read-only/Read-write	Read-only

B.26 Supervisor Module

This section refers to the Supervisor module.

B.26.1 SUPERVISOR Characteristics

Table B–89 shows the Supervisor characteristics.

Table B–89 SUPERVISOR Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.26.2 SUPERVISOR GROUP Characteristics

Table B–90 shows the SUPERVISOR GROUP characteristics.

Table B–90 SUPERVISOR GROUP Characteristics

Keyword	Class	Syntax	Default Value
FUNCTION	S/C	BACKUP	–
PRIMARIES	S/D	Routing circuit entity name(s)	{ }
SECONDARY	S/D	Routing circuit entity name	–
INVOKE TIMER	S	Range 1–65535	5
REVOKE TIMER	S	Range 1–65535	30

B.27 TCP Module

This section refers to the TCP module.

B.27.1 TCP Characteristics

Table B–91 shows the TCP characteristics.

Table B–91 TCP Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	X1.0.2
RETRANSMISSION TIMEOUT ALGORITHM	S/D	Constant	Constant
MINIMUM RETRANSMISSION TIMEOUT	S/D	Number of millisecs.	1000 millisecs
MAXIMUM RETRANSMISSION TIMEOUT	S/D	Number of millisecs.	64000 millisecs
MAXIMUM CONNECTIONS	S/D	-1 for maximum dynamic connections 1–maximum concurrent connections	-1
TIME TO LIVE	S	Range 1–255 (to satisfy RFC 1122)	64

B.28 X25 Access Module

This section refers to the X25 Access module.

Note that many of the characteristics for the X25 Access module entities are dependent on the profile of the network you are using. For details of these characteristics, refer to the online *Network Information*.

B.28.1 X25 ACCESS Characteristics

Table B–92 shows the X25 ACCESS characteristics.

Table B–92 X25 ACCESS Characteristics

Keyword	Class	Syntax	Default Value
MAXIMUM ACTIVE PORTS	S/C	Decimal number	–
VERSION	R	Version number	–

B.28.2 X25 ACCESS DTE CLASS Characteristics

Table B–93 shows the X25 ACCESS DTE CLASS characteristics.

Table B–93 X25 ACCESS DTE CLASS Characteristics

Keyword	Class	Syntax	Default Value
ACCOUNT	S	Octet String	–
DESTINATION	S	Full name	–
LOCAL DTES	S	Set of simple names	–
NODE	S	Node name	–
OUTGOING SESSION TEMPLATE	S	Simple name	Default
PASSWORD	W	Octet string	–
SECURITY DTE CLASS	S	Simple name	Default
SEGMENT SIZE	S	Range 1–4096	64
TYPE	S/C	LOCAL or REMOTE	–
USER	W	Octet string	–

B.28.3 X25 ACCESS FILTER Characteristics

Table B–94 shows the X25 ACCESS FILTER characteristics.

Table B–94 X25 ACCESS FILTER Characteristics

Keyword	Class	Syntax	Default Value
CALL DATA MASK	S	Octet String	Null string
CALL DATA VALUE	S	Octet String	Null string
CALLED ADDRESS EXTENSION MASK	S	Octet String	Null string
CALLED ADDRESS EXTENSION VALUE	S	Octet String	Null string
GROUP	S	Simple name	–
INBOUND DTE CLASS	S	Simple name	–
INCOMING DTE ADDRESS	S	DTE address. Can include wildcard.	–
ORIGINALLY CALLED ADDRESS	S	DTE address	–
PRIORITY	S	Range 0–65535	1
RECEIVING DTE ADDRESS	S	DTE address	–
REDIRECT REASON	S	One of: Busy Not specified Out of order Systematic	Not specified
SECURITY FILTER	S	Simple name	Default
SENDING DTE ADDRESS	S	DTE address	–
SUBADDRESS RANGE	S	Range of an even number of hex digits	–

B.28.4 X25 ACCESS REACHABLE ADDRESS Characteristics

Table B–95 shows the X25 ACCESS REACHABLE ADDRESS characteristics.

Table B–95 X25 ACCESS REACHABLE ADDRESS Characteristics

Keyword	Class	Syntax	Default Value
ADDRESS EXTENSIONS	S	TRUE/FALSE	TRUE
ADDRESS PREFIX	S/C	DTE address	–
DESTINATION	S	DTE address	–
DTE CLASS	S	Simple name	–
MAPPING	S	Manual/X.121	X.121

B.28.5 X25 ACCESS SECURITY DTE CLASS REMOTE DTE Characteristics

Table B–96 shows the X25 ACCESS SECURITY DTE CLASS REMOTE DTE characteristics.

Table B–96 X25 ACCESS SECURITY DTE CLASS REMOTE DTE Characteristics

Keyword	Class	Syntax	Default Value
ACL	S	Access control list	–
REMOTE ADDRESS PREFIX	S/C	DTE address	–
RIGHTS IDENTIFIERS	S	Set of simple names	–

B.28.6 X25 ACCESS SECURITY FILTER Characteristics

Table B–97 shows the X25 ACCESS SECURITY FILTER characteristics.

Table B–97 X25 ACCESS SECURITY FILTER Characteristics

Keyword	Class	Syntax	Default Value
ACL	S	Access control list	–

B.28.7 X25 ACCESS TEMPLATE Characteristics

Table B–98 shows the X25 ACCESS TEMPLATE characteristics.

Table B–98 X25 ACCESS TEMPLATE Characteristics

Keyword	Class	Syntax	Default Value
CALL DATA	S	Octet String	–
CALLING ADDRESS EXTENSION	S	NSAP address	–
CHARGING INFORMATION	S	TRUE/FALSE	FALSE
DESTINATION DTE ADDRESS	S	DTE address	–
DTE CLASS	S	Simple name	–
END-TO-END DELAY	S	Range	0..0
EXPEDITED DATA	S	One of: Do not use Not specified Use	Not specified
FAST SELECT	S	One of: Fast select No fast select Not specified With response	Not specified
LOCAL FACILITIES	S	Octet String	–
LOCAL SUBADDRESS	S	DTE address	–
NETWORK USER IDENTITY	S	Octet String	–
NSAP MAPPING	S	TRUE/FALSE	FALSE
PACKET SIZE	S	Range 0–4096	–
QUALITY OF SERVICE	S	Octet String	–
REVERSE CHARGING	S	TRUE/FALSE	FALSE
RPOA SEQUENCE	S	Set of DTE addresses	–
SELECTED GROUP	S	Simple name	–
TARGET ADDRESS EXTENSION	S	NSAP address	–
THROUGHPUT CLASS REQUEST	S	Range	0..0
TRANSIT DELAY SELECTION	S	Range	0..0
WINDOW SIZE	S	Range 0–127	–

B.29 X25 Protocol Module

This section refers to the X25 Protocol module.

B.29.1 X25 PROTOCOL Characteristic

Table B–99 shows the X25 PROTOCOL characteristic.

Table B–99 X25 PROTOCOL Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.29.2 X25 PROTOCOL DTE Characteristics

Table B–100 shows the X25 PROTOCOL DTE characteristics.

Table B–100 X25 PROTOCOL DTE Characteristics

Keyword	Class	Syntax	Default Value
CALL TIMER	S	Range 0–255	Profile-dependent
CCITT VERSION	S	Range 1–9999	1984
CLEAR TIMER	S	Range 1–255	Profile-dependent
DEFAULT PACKET SIZE	S	Range 1–4096	Profile-dependent
DEFAULT WINDOW SIZE	S/D	Range 1–127	Profile-dependent
DESCRIPTION	R	Hardware details	–
EXTENDED PACKET SEQUENCING	S	TRUE/FALSE	FALSE
INBOUND DTE CLASS	S/D	Simple Name	–
INCOMING LIST	S/D	Set of ranges of logical channel numbers	{[1..4095]}
INTERFACE TYPE	S/D	DTE, DCE	DTE
INTERRUPT TIMER	S	Range 0–255	Profile-dependent
LINK SERVICE PROVIDER	S	Local entity name	–
MAXIMUM ACTIVE CIRCUITS	S	Range 1–4096	512
MAXIMUM CLEAR ATTEMPTS	S	Range 1–255	Profile-dependent

(continued on next page)

Table B–100 (Cont.) X25 PROTOCOL DTE Characteristics

Keyword	Class	Syntax	Default Value
MAXIMUM PACKET SIZE	S	Range 16–4096	Profile-dependent
MAXIMUM RESET ATTEMPTS	S	Range 1–255	Profile-dependent
MAXIMUM RESTART ATTEMPTS	S	Range 1–255	Profile-dependent
MAXIMUM THROUGHPUT CLASSES	S	Integer	Profile-dependent
MAXIMUM WINDOW SIZE	S	Range 1–127	Profile-dependent
MINIMUM PACKET SIZE	S	Range 16–4096	Profile-dependent
MINIMUM THROUGHPUT CLASSES	S	Integer	Profile-dependent
MINIMUM WINDOW SIZE	S	Range 1–127	Profile-dependent
OUTGOING LIST	S	Set of ranges	{[1..4095]}
PROFILE	S/C	Profile name	–
RESET TIMER	S	Range 1–255	Profile-dependent
RESTART TIMER	S	Range 1–255	Profile-dependent
SEGMENT SIZE	S	Range 1–4096	64
X25 ADDRESS	S	DTE address	–

B.29.3 X25 PROTOCOL DTE PVC Characteristics

Table B–101 shows the X25 PROTOCOL DTE PVC characteristics.

Table B–101 X25 PROTOCOL DTE PVC Characteristics

Keyword	Class	Syntax	Default Value
ACL	S	Access control list	–
CHANNEL	S/C	Range 0–4095	–
PACKET SIZE	S/C	Range 16–4096	–
WINDOW SIZE	S/C	Range 1–127	–

B.29.4 X25 PROTOCOL GROUP Characteristics

Table B–102 shows the X25 PROTOCOL DTE PVC characteristics.

Table B–102 X25 PROTOCOL GROUP Characteristics

Keyword	Class	Syntax	Default Value
MEMBERS	S	Set of records	–
REMOTE DTE ADDRESS	S	DTE address	–
TYPE	S	CUG or BCUG	BCUG

B.30 X25 Relay Module

This section refers to the X25 Relay module.

B.30.1 X25 RELAY CLIENT Characteristics

Table B–103 shows the X25 RELAY CLIENT characteristics.

Table B–103 X25 RELAY CLIENT Characteristics

Keyword	Class	Syntax	Default Value
DTE CLASS	S	Simple name	–
FILTERS	S	Set of simple names	{ }
RIGHTS IDENTIFIERS	S	Set of simple names	{ }
TEMPLATE	S	Simple name	–

B.30.2 X25 RELAY PVC Characteristics

Table B–104 shows the X25 RELAY PVC characteristics.

Table B–104 X25 RELAY PVC Characteristics

Keyword	Class	Syntax	Default Value
LOCAL PVC	S	Simple name	–
RELAYED PVC	S	Simple name	–
REMOTE DTE CLASS	S	Simple name	–
RETRY LIMIT	S	Range 0–65535	10
RETRY TIMER	S	Range 1–65535	60
RIGHTS IDENTIFIERS	S	Set of simple names	{ }

B.31 X25 Server Module

This section refers to the X25 Server module.

B.31.1 X25 SERVER Characteristics

Table B–105 shows the X25 SERVER characteristics.

Table B–105 X25 SERVER Characteristics

Keyword	Class	Syntax	Default Value
INCOMING SESSION TEMPLATE	S/C	Simple name	–
MAXIMUM SESSION CONNECTIONS	S/C	Decimal number	–
VERSION	R	Version number	–

B.31.2 X25 SERVER CLIENT Characteristics

Table B–106 shows the X25 SERVER CLIENT characteristics.

Table B–106 X25 SERVER CLIENT Characteristics

Keyword	Class	Syntax	Default Value
ACCOUNT	S	Octet string	–
APPLICATION	S	Octet string	36
DESTINATION	S	End user specification	Number = 36
FILTERS	S	Set of simple names	{ }
NODE	S	Node name	–
OUTGOING SESSION TEMPLATE	S	Simple name	Default
PASSWORD	W	Octet string	–
USER	W	Octet string	–

B.31.3 X25 SERVER SECURITY NODES Characteristics

Table B–107 shows the X25 SERVER SECURITY NODES characteristics.

Table B–107 X25 SERVER SECURITY NODES Characteristics

Keyword	Class	Syntax	Default Value
NODES	S	Set of full names	{ }
RIGHTS IDENTIFIERS	S	Set of simple names	{ }

C

NCL Commands to Set Up Event Sinks

This appendix provides the NCL commands you should enter at a host system to enable it to receive events from the DECNIS. It describes how to log events to a terminal; to log events to a file, see the relevant network management documentation for the sink node.

C.1 Setting Up an OpenVMS System as an Event Sink

1. Log on to the host OpenVMS system, and start NCL.
2. Enter the following NCL commands:

   ```
   NCL> CREATE EVENT DISPATCHER
   NCL> ENABLE EVENT DISPATCHER

   NCL> CREATE EVENT DISPATCHER SINK sink-name
   NCL> SET EVENT DISPATCHER SINK sink-name END USER NUMBER = 82
   NCL> ENABLE EVENT DISPATCHER SINK sink-name

   NCL> EXIT
   $ REPLY/ENABLE
   ```

   Events will now be logged to the network operator terminal of the OpenVMS system.

C.2 Setting Up a Digital UNIX System as an Event Sink

1. Log on to the host system using the terminal to which you want events to be logged.
2. Discover the identity of the terminal by entering the following command:

   ```
   # who am i
   ```

   This returns the identity of the terminal, for example, tty10.
3. Start NCL on the system.

4. Enter the following NCL commands:

```
ncl> create event dispatcher
ncl> enable event dispatcher
```

```
ncl> create event dispatcher sink sink-name
ncl> set event dispatcher sink sink-name client type device
ncl> set event dispatcher sink sink-name device name\
_ncl> /dev/terminal-id
```

   where *terminal-id* is the identity of the terminal as returned in step 2, for example, tty10.

```
ncl> enable event dispatcher sink sink-name
```

D

Common Protocol Values and Formats

This appendix lists protocol values that appear in the various types of frames that the bridging functions of the DECNIS allows you to filter or forward.

- Ethernet Protocol Type Values
- IEEE 802.2 LSAP Protocol Values
- IEEE 802.2 SNAP Protocol IDs

D.1 Ethernet Protocol Type Values

This section lists Ethernet Protocol Type values. These are 2-byte hexadecimal values in the form *nn-nn* that appear in the Protocol Type field of Ethernet frames. They identify the higher-layer protocol information contained in the frame.

This section contains the following tables:

- Public Ethernet Protocol Types
- Digital Ethernet Protocol Types
- Publicly Available Private Ethernet Protocol Types

D.1.1 Public Ethernet Protocol Types

Table D–1 lists the public Ethernet Protocol Type values.

Table D–1 Public Ethernet Protocol Types

Protocol Value	Description
01-01 to 01-FF	Experimental
06-00	Xerox NS Internet (XNS)

(continued on next page)

Table D–1 (Cont.) Public Ethernet Protocol Types

Protocol Value	Description
08-00	DoD Internet (TCP/IP)
08-01	X.75 Internet
08-02	NBS Internet
08-03	ECMA Internet
08-04	CHOASnet, proposed by Symbolics, Inc.
08-05	X.25 Level 3
08-06	Ethernet Address Resolution Protocol (ARP)
08-1C	Proposed by Symbolics, Inc.
80-06	Nestar
90-00	Loopback

D.1.2 Digital Ethernet Protocol Types

Table D–2 lists the Ethernet Protocol Types assigned by Digital from its block of values.

Table D–2 Digital Ethernet Protocol Types

Protocol Value	Type	Description
60-01	DEC_ MOPDL	DNA Dump/Load (MOP)
60-02	DEC_ MOPRC	DNA Remote Console (MOP)
60-03	DEC_NET	DNA Routing
60-04	DEC_LAT	Local Area Transport (LAT)
60-05	DEC_DIAG	Diagnostics
60-06	DEC_C_USE	Customer use
60-07	DEC_LAVC	System Communication Architecture (SCA)
80-38	DEC_ BRIDGE	DEC Spanning Tree
80-3B		VAXELN™ software

(continued on next page)

Table D–2 (Cont.) Digital Ethernet Protocol Types

Protocol Value	Type	Description
80-3C		DNA Naming Service
80-3D		CSMA/CD Encryption
80-3F	DEC_LTM	LAN Traffic Monitor
80-40		NetBIOS emulator (PCSG)
80-41		Pathworks™ (LAD/LAST)
80-42		Reserved

D.1.3 Publicly Available Private Ethernet Protocol Types

Table D–3 lists Ethernet Protocol Type values held by the organizations that decided to make this information publicly available. Refer to the organization's documentation for the codes assigned to specific protocols.

Table D–3 Publicly Available Private Ethernet Protocol Types

Protocol Value	Organization Name	Location
04-00	Nixdorf Computer, AG	Fuerstenaller, West Germany
08-88 to 08-8A	Xyplex	Concord, MA
09-00	Ungermann-Bass® (Network Debugger)	California
0A-00	Xerox Corp.	Stamford, CT
0B-AD	Banyan Systems	Westboro, MA
10-00	Berkeley Trailer (Negotiation)	Berkeley, CA
10-01 to 10-0F	Berkeley Trailer (Encapsulation, IP Trailer Block)	Berkeley, CA
52-08	Bolt Beranek and Newman Inc. (Simnet)	Cambridge, MA
60-10 to 60-14	3Com Corp.	Santa Clara, CA
70-00	Ungermann-Bass (Download)	California
70-01 to 70-02	Ungermann-Bass (NIU)	California
70-03	Interlan	Boxborough, MA

(continued on next page)

Table D–3 (Cont.) Publicly Available Private Ethernet Protocol Types

Protocol Value	Organization Name	Location
70-20 to 70-29	LRT	Reading, Berks, England
70-30	Proteon	Natick, MA
80-03	Cronus Industries, Inc. (VLN)	Dallas, TX
80-04	Cronus Industries, Inc. (Direct)	Dallas, TX
80-05	Hewlett-Packard Co.	Palo Alto, CA
80-06	Nestar	San Jose, CA
80-08	American Telephone & Telegraph Co.	New York, NY
80-10	Excelan, Inc.	San Jose, CA
80-13	Silicon Graphics Inc. (Diagnostics)	Mountain View, CA
80-14	Silicon Graphics Inc. (Network Games)	Mountain View, CA
80-15	Silicon Graphics Inc. (reserved)	Mountain View, CA
80-16	Silicon Graphics Inc. (XNS Name Server)	Mountain View, CA
80-19	Apollo Computer Inc., sub. of Hewlett-Packard Co.	Chelmsford, MA
80-2E	Tymshare	Cupertino, CA
80-2F	Tigan, Inc.	Palo Alto, CA
80-35	Stanford University (Reverse ARP)	Stanford, CA
80-36	Aeonic Systems	Billerica, MA
80-44	Planning Research Corp., sub. of Emhart PRC	McLean, VA
80-46 to 80-47	American Telephone & Telegraph Co.	New York, NY
80-49	Expert Data	Boulogne, France
80-5B to 80-5C	Stanford University (V System)	Stanford, CA

(continued on next page)

Table D–3 (Cont.) Publicly Available Private Ethernet Protocol Types

Protocol Value	Organization Name	Location
80-5D	Evans & Sutherland Computer Corp.	Salt Lake City, UT
80-60	Little Machines	San Diego, CA
80-62	Counterpoint Computers	San Jose, CA
80-65 to 80-66	University of Massachusetts	Amherst, MA
80-67	Veeco Integrated Automation	Dallas, TX
80-68	General Dynamics Corp., Fort Worth Div.	Fort Worth, TX
80-69	American Telephone & Telegraph Co.	New York, NY
80-6A	Autophon	Solothurn, Switzerland
80-6C	ComDesign	Goleta, CA
80-6D	Compugraphic Corp., sub. of Bayer AG	Wilmington, MA
80-6E to 80-77	Landmark Graphics Corp.	Houston, TX
80-7A	Matra	Paris, France
80-7B	Dansk Data Elektronik A/S	Herlev, Denmark
80-7C	University of Michigan	Ann Arbor, MI
80-7D to 80-7F	Vitalink Communications Corp.	Fremont, CA
80-80	Vitalink Communications Corp. (Bridge Management)	Fremont, CA
80-81 to 80-83	Counterpoint Computers	San Jose, CA
80-9B	Kinetics (AppleTalk)	Walnut Creek, CA
80-9C to 80-9E	Datability	New York, NY
80-9F	Spider Systems Ltd.	Edinburgh, Scotland
80-A3	Nixdorf Computer, A.G.	Fuerstenaller, West Germany
80-A4 to 80-B3	Siemens Gammasonics, sub. of Siemens A.G.	Des Plaines, IL
80-C0 to 80-C3	Digital Communications Associates Inc.	Alpharetta, GA

(continued on next page)

Table D–3 (Cont.) Publicly Available Private Ethernet Protocol Types

Protocol Value	Organization Name	Location
80-C6	Pacer Software	La Jolla, CA
80-C7	Applitek Corp.	Wakefield, MA
80-C8 to 80-CC	Intergraph Corp.	Huntsville, AL
80-CD to 80-CE	Harris Corp.	Melbourne, FL
80-CF to 80-D2	Taylor Instruments	Rochester, NY
80-D3 to 80-D4	Rosemont Corp.	La Habra, CA
80-D5	Ungermann-Bass	California
80-DD	Varian Associates	Palo Alto, CA
80-DE to 80-DF	Integrated Solutions (Transparent Remote File System)	San Jose, CA
80-E0 to 80-E3	Allen Bradley Co., sub. of Rockwell International Corp.	Ann Arbor, MI
80-E4 to 80-F0	Datability	New York, NY
80-F2	Retix	Santa Monica, CA
80-F3	Kinetics (AppleTalk Address Resolution Protocol)	Walnut Creek, CA
80-F4 to 80-F5	Kinetics	Walnut Creek, CA
80-F7	Apollo Computer Inc., sub. of Hewlett-Packard Co.	Chelmsford, MA
80-FF to 81-03	Wellfleet Communications	Bedford, MA
81-07 to 81-09	Symbolics, Inc.	Cambridge MA
81-30	Waterloo Manufacturing Co., Ltd.	Waterloo, Ontario, Canada
81-31	VG Laboratory Systems	Cheshire, England
81-37 to 81-38	Novell, Inc.	Provo, UT
81-39 to 81-3D	KTI	San Jose, CA
90-01	Bridge, Inc. (Bridge Management)	Cupertino, CA
90-02	Bridge, Inc. (Terminal Servers)	Cupertino, CA

(continued on next page)

Table D–3 (Cont.) Publicly Available Private Ethernet Protocol Types

Protocol Value	Organization Name	Location
AF-AF	Logicraft	
FF-00	Bolt Beranek and Newman Inc. (VITAL-LANBridge)	Cambridge, MA

D.2 IEEE 802.2 LSAP Protocol Values

Table D–4 lists the IEEE 802.2 LSAP protocol values. These values appear in the DSAP (Destination Service Access Point) and SSAP (Source Service Access Point) fields of IEEE 802.3 frames, identifying the higher-layer protocol contained in the frame. IEEE 802.2 LSAP protocol values are one hexadecimal byte in the form *nn*.

The LSAP (DSAP and SSAP) values are assigned by the IEEE; each one identifies a particular protocol defined by a national or international standard. For example, an IEEE 802.3 frame with FE in its SSAP and DSAP fields contains higher-layer information that conforms to the ISO Connectionless-Mode Network Protocol (ISO 8473).

Table D–4 LSAP Protocol Values

Protocol Value	Description
00	Null LSAP
02	Individual Logical Link Control (LLC) Sublayer Management Function
03	Group Logical Link Control (LLC) Sublayer Management Function
06	Transmission Control Protocol/Internet Protocol (TCP/IP)
0E	PROWAY Network Management and Initialization
42	Bridge Spanning Tree Protocol
4E	Manufacturing Message Service
8E	PROWAY-LAN
AA	Sub-Network Access Protocol (SNAP)
FE	ISO Connectionless-Mode Network Protocol
FF	Global LSAP

D.3 IEEE 802.2 SNAP Protocol IDs

Table D–5 lists the IEEE 802.2 SNAP Protocol IDs assigned by Digital Equipment Corporation. For SNAP Protocol IDs assigned by other organizations, refer to the organization's documentation or contact the organization directly.

SNAP Protocol IDs identify the higher-layer protocol contained in a frame in the case where the protocol is not a national or international standard. In an IEEE 802.2 SNAP frame, the first 5 bytes after the control field form the Protocol-ID field. The Protocol-ID value is 5 bytes in the form *nn-nn-nn-nn-nn*. The first 3 bytes are taken from the 24-bit organization unique identifier assigned by the IEEE; the last 2 bytes are assigned by the owner of the organization unique identifier. For example, an IEEE 802.2 SNAP frame with the value 08-00-2B-80-3C in its Protocol-ID field contains a DNA Naming Service protocol message (08-00-2B is the organization unique identifier for Digital).

Table D–5 Digital IEEE 802.2 SNAP Protocol Identification Assignments

Protocol Identification Code	Assignment
08-00-2B-60-01	DNA Dump/Load (MOP)
08-00-2B-60-02	DNA Remote Console (MOP)
08-00-2B-60-03	DNA Routing
08-00-2B-60-04	Local Area Transport (LAT)
08-00-2B-60-05	Diagnostics
08-00-2B-60-06	Customer use
08-00-2B-60-07	System Communication Architecture (SCA)
08-00-2B-80-3B	VAXELN software
08-00-2B-80-3C	DNA Naming Service
08-00-2B-80-3D	CSMA/CD Encryption
08-00-2B-80-3F	LAN Traffic Monitor
08-00-2B-80-40	NetBIOS emulator (PCSG)
08-00-2B-90-00	MOP LAN Loopback protocol

E

Characteristic Values of the Default X25 ACCESS Template

Table E–1 shows the values of the characteristics in the ‘default’ template.

Table E–1 Characteristic Values of ‘Default’ Template

Characteristic	Value in ‘Default’ Template
DTE class	Not specified
Call data	Not specified
Packet size	Not specified
Window size	Not specified
Destination X.25 address	Not specified
Fast select option	Not specified
Reverse charging	False
Selected Group	Not specified
Throughput class request	Not specified
Network user identity	Not specified
Local facilities	Not specified
Charging information	False
RPOA sequence	Not specified
Local subaddress	Not specified
Target address extension	Not specified
NSAP mapping	False
Calling address extension	Not specified
Transit delay selection	Not specified

(continued on next page)

Table E–1 (Cont.) Characteristic Values of ‘Default’ Template

Characteristic	Value in ‘Default’ Template
End-to-end delay	Not specified
Quality of service	Not specified
Expedited data option	Not specified

F

Connectivity Tables

This appendix contains routing information required for connecting to a DECnet/OSI or DECnet Phase IV system. You can use it to check:

- Area restrictions
- Whether you need to use manual routing
- Which network protocols you should specify for PPP circuits

F.1 Where to Start

Check whether the remote system is a DECnet/OSI or DECnet Phase IV system:

- If the remote system is a DECnet/OSI system only, for example, another DECNIS, refer to Section F.2.
- If the remote system is a DECnet Phase IV system only, refer to Section F.3.
- If the remote system is another vendor's system that runs DECnet/OSI and DECnet Phase IV at the same level and as independent protocols, you cannot connect to it from the DECNIS.

  Some vendor systems can be configured in this way so that DECnet/OSI and DECnet Phase IV are not integrated but are run independently and simultaneously.

  If possible, reconfigure the remote system to use either DECnet/OSI or DECnet Phase IV.

F.2 Checking Information Required for Connections to DECnet/OSI Systems

F.2.1 Procedure

Check the following table for details of the routing algorithms used by the DECNIS and take the corresponding action:

If DECNIS is using...	Then check...
Level 1 link state Level 2 link state	**Table F–1** for the routing algorithms used by the remote system and refer to the corresponding connectivity information.
Level 1 routing vector Level 2 link state	**Table F–2** for the routing algorithms used by the remote system and refer to the corresponding connectivity information.
Level 1 link state Level 2 routing vector	**Table F–3** for the routing algorithms used by the remote system and refer to the corresponding connectivity information.
Level 1 routing vector Level 2 routing vector	**Table F–4** for the routing algorithms used by the remote system and refer to the corresponding connectivity information.

Table F–1 DECNIS Uses Link State at Level 1 and Link State at Level 2

	Connectivity Information		
Algorithms Used by Remote System	Area Restrictions	Manual Routing	Network Protocols†
Level 1 link state Level 2 link state	DECNIS and remote system can be in the same area or in different areas	No	ISO8473
Level 1 routing vector Level 2 link state	DECNIS and remote system must be in different areas	No	ISO8473
Level 1 link state Level 2 routing vector	DECNIS and remote system must be in different areas	Yes‡	ISO8473, DECNET PHASE IV
Level 1 routing vector Level 2 routing vector	DECNIS and remote system must be in different areas	Yes‡	ISO8473, DECNET PHASE IV

†PPP circuits only. See Section 9.8 for more information.

‡For more information about manual routing, see Section 16.15.

Table F–2 DECNIS Uses Routing Vector at Level 1 and Link State at Level 2

	Connectivity Information		
Algorithms Used by Remote System	**Area Restrictions**	**Manual Routing**	**Network Protocols†**
Level 1 link state Level 2 link state	DECNIS and remote system must be in different areas	No	ISO8473
Level 1 routing vector Level 2 link state	DECNIS and remote system can be in the same area or in different areas	No	ISO8473
Level 1 link state Level 2 routing vector	DECNIS and remote system must be in different areas	Yes‡	ISO8473, DECNET PHASE IV
Level 1 routing vector Level 2 routing vector	DECNIS and remote system must be in different areas	Yes‡	ISO8473, DECNET PHASE IV

†PPP circuits only. See Section 9.8 for more information.

‡For more information about manual routing, see Section 16.15.

Table F–3 DECNIS Uses Link State at Level 1 and Routing Vector at Level 2

	Connectivity Information		
Algorithms Used by Remote System	Area Restrictions	Manual Routing	Network Protocols†
Level 1 link state Level 2 link state	DECNIS and remote system must be in different areas	Yes‡	ISO8473, DECNET PHASE IV
Level 1 routing vector Level 2 link state	DECNIS and remote system must be in different areas	Yes‡	ISO8473, DECNET PHASE IV
Level 1 link state Level 2 routing vector	DECNIS and remote system can be in the same area or in different areas	No	ISO8473, DECNET PHASE IV
Level 1 routing vector Level 2 routing vector	DECNIS and remote system must be in different areas	No	ISO8473, DECNET PHASE IV

†PPP circuits only. See Section 9.8 for more information.

‡For more information about manual routing, see Section 16.15.

Table F–4 DECNIS Uses Routing Vector at Level 1 and Routing Vector at Level 2

	Connectivity Information		
Algorithms Used by Remote System	Area Restrictions	Manual Routing	Network Protocols†
Level 1 link state Level 2 link state	DECNIS and remote system must be in different areas	Yes‡	ISO8473, DECNET PHASE IV
Level 1 routing vector Level 2 link state	DECNIS and remote system must be in different areas	Yes‡	ISO8473, DECNET PHASE IV
Level 1 link state Level 2 routing vector	DECNIS and remote system must be in different areas	No	ISO8473, DECNET PHASE IV
Level 1 routing vector Level 2 routing vector	DECNIS and remote system can be in the same area or in different areas	No	ISO8473, DECNET PHASE IV

†PPP circuits only. See Section 9.8 for more information.

‡For more information about manual routing, see Section 16.15.

F.3 Checking Information Required for Connections to DECnet Phase IV Systems

F.3.1 Procedure

A DECnet Phase IV system runs:

- Routing vector algorithm at Level 1
- Routing vector algorithm at Level 2

Check Table F–5 for details of the routing algorithms used by the DECNIS and refer to the corresponding connectivity information.

Table F–5 Information for Connecting to DECnet Phase IV Systems

	Connectivity Information		
Algorithms Used by DECNIS	**Area Restrictions**	**Manual Routing**	**Network Protocols†**
Level 1 link state Level 2 link state	DECNIS and remote system must be in different areas	Yes‡	DECNET PHASE IV
Level 1 link state Level 2 routing vector	DECNIS and remote system must be in different areas	No	DECNET PHASE IV
Level 1 routing vector Level 2 link state	DECNIS and remote system must be in different areas	Yes‡	DECNET PHASE IV
Level 1 routing vector Level 2 routing vector	DECNIS and remote system can be in the same area or in different areas	No	DECNET PHASE IV

†PPP circuits only. See Section 9.8 for more information.

‡For more information about manual routing, see Section 16.15.

G
Example NCL Scripts

This appendix contains the NCL commands required to accomplish various management tasks. Where required, a brief explanation of the command is provided.

In addition to the examples in this appendix, there are further example NCL scripts in the following directories:

- On MS–DOS PCs and Windows 95/NT PCs

  install-directory\ DOCS

  where *install-directory* is the installation directory.

- On OpenVMS systems:

  SYS$EXAMPLES

- On Digital UNIX systems:

  /usr/lib/dnet

G.1 Creating a CSMA/CD Routing Circuit for DECnet/OSI and IP (RIP) Routing

This example illustrates the NCL commands required to set up a CSMA/CD connection to the network, and configure it for DECnet/OSI routing and IP routing using the RIP protocol.

```
create routing type L1Router  , protocols { ISO8473,IP } ❶
add routing manual area address { 37:23:1 } ❷
set routing manual L1 algorithm link state ❸
set routing maximum path splits 1 ❹
set routing autonomous system number 124 ❺
enable routing

create device unit csmacd_device1 name L601-3 ❻
enable device unit csmacd_device1

create csma-cd station csmacd_station1 communication port L601-3-0 ❼
enable csma-cd station csmacd_station1

create routing circuit csmacd-0 type csma-cd ❽
set routing circuit csmacd-0 data link entity csma-cd station
    csmacd_station1 ❾
set routing circuit csmacd-0 originating queue limit 32 ❿
set routing circuit csmacd-0 L1 cost 20 ⓫
set routing circuit csmacd-0 L1 router priority 64 ⓬

set routing circuit csmacd-0 subnet address { address= 12.23.2.4, -
    mask = 255.255.0.0 } ⓭

enable routing circuit csmacd-0

create routing network protocol ip ⓮
enable routing network protocol ip

create routing control protocol rip-1 type ip rip ⓯
set routing control protocol rip-1 ip subnet {{address = 12.23.2.4, -
    mask = 255.255.0.0}} ⓰
set routing control protocol rip-1 rip state send and receive ⓱
enable routing control protocol rip-1
```

❶ Create the Routing module, and specify that router will be a Level 1 router; see Section 9.24. Set the protocols to include both IP and OSI.

❷ Set the Phase V area address; see Section 16.7.

❸ Set the routing algorithm to link state; see Section 16.9.

❹ Disable path splitting; that is, prevent two or more equal cost paths between two nodes from sharing traffic.

❺ Specify the Autonomous System Number of your system. This is a number assigned by the DDN Network Information Center. You must do this if the DECNIS will use the EGP protocol.

❻ The CSMA/CD connection will use a LANcontroller 601 Network Interface Card, in slot 3 of the backplane. Create and enable a device unit for this Network Interface Card; see Section 8.4.

❼ Create and enable a CSMA/CD data link on the port provided by the LANcontroller 601 Network Interface Card; see Section 8.7.2.

❽ Create the routing circuit and specify its type.

❾ Specify that the routing circuit is to use the CSMA/CD station created above as its data link.

❿ Set the originating queue limit; see Section 9.3.

⓫ Set the Level 1 cost; see Section 16.11.

⓬ Set the Level 1 routing priority; see Section 16.12.

⓭ Set the IP address and subnet mask for the interface; see Section 10.5.

⓮ Create a NETWORK PROTOCOL entity of type IP.

⓯ Create a ROUTING CONTROL entity of type RIP.

⓰ Specify the subnet address to which this entity applies. This is the same as the subnet address of the routing circuit.

⓱ Set the RIP state to allow RIP messages to be sent and received over this circuit; see Section 10.10.3.

G.2 Creating an FDDI Routing Circuit for DECnet/OSI, IP (IS–IS), AppleTalk, and NetWare IPX Routing

This example illustrates the NCL commands required to set up an FDDI connection to the network, and configure it for DECnet/OSI routing and IP routing using the IS–IS protocol. It will also be used for native routing of NetWare IPX and AppleTalk.

```
create routing type L1Router  , protocols { ISO8473,IP } ❶
add routing manual area address { 37:23:1 } ❷
set routing manual L1 algorithm link state ❸
enable routing

create device unit fddi_device1 name F621-6 ❹
enable device unit fddi_device1

create fddi ❺
create fddi station fddi_station1 communication port F621-6-0 ❻

create mop circuit fddi-mop-0 type fddi ❼
set mop circuit fddi-mop-0 link name fddi station fddi_station1 ❽
enable mop circuit fddi-mop-0 function {loop requester} ❾

create routing circuit fddi-0 type fddi ❿
set routing circuit fddi-0 data link entity fddi station
    fddi_station1 ⓫
set routing circuit fddi-0 originating queue limit 32 ⓬
set routing circuit fddi-0 L1 cost 20 ⓭
set routing circuit fddi-0 L1 router priority 64 ⓮
set routing circuit csmacd-0 subnet address { address= 12.23.2.5, -
    mask= 255.255.0.0 } ⓯

create routing network protocol appletalk⓰
enable routing network protocol appletalk
create routing network protocol netware ipx
enable routing network protocol netware ipx

create routing circuit fddi-0 network protocol appletalk⓱
create routing circuit fddi-0 network protocol netware ipx

set routing circuit fddi-0 network protocol appletalk - ⓲
   appletalk manual network address {network = 1500, node = 226}

set routing circuit fddi-0 network protocol netware ipx - ⓳
   netware network number %x223f

enable routing circuit fddi-0 network protocol appletalk
enable routing circuit fddi-0 network protocol netware ipx

enable fddi station fddi_station1 ⓴


enable routing circuit fddi-0
```

❶ Create the Routing module, and specify that router will be a Level 1 router; see Section 9.24. Set the protocols to include both IP and OSI.

❷ Set the Phase V area address; see Section 16.7.

❸ Set the routing algorithm to link state; see Section 16.9.

❹ The FDDI connection will use an FDDIcontroller 621 Network Interface Card, in slots 5 and 6 of the backplane. Create and enable a device unit for this Network Interface Card; see Section 8.4.

❺ Create the FDDI entity.

❻ Create an FDDI data link on the port provided by the FDDIcontroller 621 Network Interface Card; see Section 8.8.

❼ Create a MOP circuit on this FDDI data link, so that loopback testing can be carried out; see Section 8.26.

❽ Associate the MOP circuit with the FDDI data link.

❾ Allow the MOP circuit to be used for loopback testing.

❿ Create the routing circuit and specify its type.

⓫ Specify that the routing circuit is to use the FDDI station created above as its data link.

⓬ Set the originating queue limit; see Section 9.21.3.

⓭ Set the Level 1 cost; see Section 16.11.

⓮ Set the Level 1 routing priority; see Section 16.12.

⓯ Set the IP address and subnet mask for the interface; see Section 10.5.

⓰ Allow the DECNIS to route AppleTalk and NetWare IPX.

⓱ Configure the routing circuit to route AppleTalk and NetWare IPX.

⓲ Set the AppleTalk address for the circuit.

⓳ Set the NetWare IPX address for the circuit.

⓴ Only enable the FDDI station after setting up the ROUTING CIRCUIT NETWORK PROTOCOL entities.

G.3 Creating an HDLC Routing Circuit for DECnet/OSI and IP (IS–IS) Routing

This example illustrates the NCL commands required to set up an HDLC connection to the network, and configure it for DECnet/OSI routing and IP routing using the IS–IS protocol.

```
create routing type L2Router  , protocols { ISO8473,IP } ❶
add routing manual area address { 37:23: } ❷
set routing manual L1 algorithm link state ❸
set routing manual L2 algorithm link state
enable routing

create device unit synch_device1 name W618-4 ❹
enable device unit synch_device1

create modem connect line hdlc_line1 communication port W618-4-0, -
    profile  "NORMAL" ❺
set modem connect line hdlc_line1 speed 64000 ❻
set modem connect line hdlc_line1 clock external ❼
set modem connect line hdlc_line1 modem control full ❽
set modem connect line hdlc_line1 suppress test indicator true ❾
enable modem connect line hdlc_line1

create hdlc link hdlc_link1 linktype balanced ❿
set hdlc link hdlc_link1 physical line modem connect line hdlc_line1 ⓫
set hdlc link hdlc_link1 preferred window size 8 ⓬
set hdlc link hdlc_link1 acknowledge timer 3000 ⓭
set hdlc link hdlc_link1 holdback timer 300
enable hdlc link hdlc_link1

create hdlc link hdlc_link1 logical station hdlc_stat1
enable hdlc link hdlc_link1 logical station hdlc_stat1 ⓮

create routing circuit hdlc_circ1 type hdlc ⓯
set routing circuit hdlc_circ1 data link entity hdlc link hdlc_link1 -
    logical station hdlc_stat1 ⓰
set routing circuit hdlc_circ1 Manual L2Only Mode FALSE ⓱
set routing circuit hdlc_circ1 L1 cost 20 ⓲
set routing circuit hdlc_circ1 L2 cost 20
set routing circuit hdlc_circ1 transmit verifier %xafafaf ⓳
set routing circuit hdlc_circ1 receive verifier %xfafafa ⓴
set routing circuit hdlc_circ1 subnet address { address= 12.23.34.45, -
    mask= 255.255.0.0 } ㉑

enable routing circuit hdlc_circ1
```

❶ Create the Routing module, and specify that router will be a Level 2 router; see Section 9.24. Set the protocols to include both IP and OSI.

❷ Set the Phase V area address; see Section 16.7.

❸ Set the routing algorithm to link state at both Level 1 and Level 2; see Section 16.9.

❹ The HDLC connection will use a DEC WANcontroller 618 Network Interface Card, in slot 4 of the backplane. Create and enable a device unit for this Network Interface Card; see Section 8.4.

❺ The HDLC routing circuit will use port 0 of the WANcontroller 618 Network Interface Card.

❻ Set the line speed to 64 kbits/s.

❼ The modem supplies the clock on this line.

❽ The line will take note of all the modem control signals; see Section 8.10.

❾ The Test Mode signal on the line will not be monitored.

❿ Create the HDLC data link; see Section 8.10.

⓫ Tell the HDLC data link to use the line created above.

⓬ Set the frame-level window size for this link; see Section 8.12.

⓭ See Section 8.11.

⓮ Create the HDLC logical station; see Section 8.10.

⓯ Create the routing circuit that is to use the HDLC data link; see Section 9.5.

⓰ Specify the data link that the routing circuit is to use; see Section 9.5.

⓱ The routing circuit can be used for both Level 1 and Level 2 traffic; see Section 9.25.

⓲ Set the circuit costs for Level 1 and Level 2 traffic; see Section 16.11.

⓳ The adjacent system to which this HDLC routing circuit connects requires to receive a verifier of AFAFAF; see Section 9.26.4.

⓴ The DECNIS requires to receive a verifier of FAFAFA from the adjacent system; see Section 9.26.4.

㉑ Set the IP address and subnet mask for the IP interface.

G.4 Creating a PPP Routing Circuit (Over Frame Relay) for DECnet/OSI and IP (IS–IS) Routing

This example illustrates the NCL commands required to set up a PPP connection to the network, and configure it for DECnet/OSI routing and IP routing using the IS–IS protocol.

The example assumes that the Routing entity has been set up as in Section G.3; the routing circuit will be set up on the WANcontroller 618 card configured in Section G.3.

```
create device unit synch_device2 name W622-7-0 ❶
enable device unit synch_device2

create modem connect line ppp_line1 communication port W622-7-0, -
    profile  "NORMAL" ❷
set modem connect line ppp_line1 clock external ❸
set modem connect line ppp_line1 modem control full ❹
set modem connect line ppp_line1 suppress test indicator true ❺
enable modem connect line ppp_line1

create frbs
create frbs channel fr_chan1 specification joint ❻
set frbs channel fr_chan1 physical line modem connect line ppp_line1 ❼
enable frbs channel fr_chan1

create frbs channel fr_chan1 connection fr_chan1_conn1 ❽
set frbs channel fr_chan1 connection fr_chan1_conn1 ❾
    preferred dlci 2598
enable frbs channel fr_chan1 connection fr_chan1_conn1

create ppp
create ppp link ppp_link1 type synchronous ❿
set ppp link ppp_link1 lower layer entity frbs channel ⓫
    fr_chan1 connection fr_chan1_conn1
enable ppp link ppp_link1

create routing circuit ppp_circ1 type ppp ⓬
set routing circuit ppp_circ1 data link entity ppp link ppp_link1 ⓭
set routing circuit ppp_circ1 Manual L2Only Mode FALSE ⓮
set routing circuit ppp_circ1 L1 cost 20 ⓯
set routing circuit ppp_circ1 L2 cost 20
set routing circuit ppp_circ1 transmit verifier %xafafaf ⓰
set routing circuit ppp_circ1 receive verifier %xfafafa ⓱
set routing circuit ppp_circ1 subnet address { address= 12.23.34.46, -
    mask= 255.255.0.0 } ⓲

enable routing circuit ppp_circ1
```

❶ Create a device unit for the WANcontroller 622 card.

❷ The PPP routing circuit will use port 0 of the WANcontroller 622 Network Interface Card.

❸ The modem supplies the clock on this line.

❹ The line will take note of all the modem control signals; see Section 8.10.

❺ The Test Mode signal on the line will not be monitored.

❻ Create the frame relay channel; the frame relay network to which the DECNIS is connected uses the Joint management protocol.

❼ Associate this channel with the line created above.

❽ Create a frame relay connection on this channel.

❾ Identify the PVC (as advised by the frame relay network) associated with this connection.

❿ Create the PPP data link; see Section 8.18.

⓫ Tell the PPP data link to use the frame relay connection established above.

⓬ Create the routing circuit that is to use the PPP data link; see Section 9.9.2.

⓭ Specify the data link that the routing circuit is to use; see Section 9.9.2.

⓮ The routing circuit can be used for both Level 1 and Level 2 traffic; see Section 9.25.

⓯ Set the circuit costs for Level 1 and Level 2 traffic; see Section 16.11.

⓰ The adjacent system to which this PPP routing circuit connects requires to receive a verifier of AFAFAF; see Section 9.26.4.

⓱ The DECNIS requires to receive a verifier of FAFAFA from the adjacent system; see Section 9.26.4.

⓲ Set the IP address and subnet mask for the IP interface.

G.5 Creating Tunnel Circuits for AppleTalk and NetWare IPX Routing

This example illustrates the NCL commands to set up a point-to-point tunnel circuit to route both AppleTalk and NetWare IPX packets, and a broadcast tunnel circuit to route NetWare IPX packets.

This example assumes that the DECNIS is already configured as an IP router, and that the ROUTING NETWORK PROTOCOL entities for both AppleTalk and NetWare IPX exist.

```
create routing circuit tunnel_point type = virtual point to point❶

create routing circuit tunnel_point network protocol appletalk❷

create routing circuit tunnel_point network protocol netware ipx
set routing circuit tunnel_point network protocol netware ipx
   netware network number %x234f

set routing circuit encapsulation destination ip address 12.34.56.78❸

enable routing circuit tunnel_point❹
enable routing circuit tunnel_point network protocol appletalk
enable routing circuit tunnel_point network protocol netware ipx

create routing circuit tunnel_broad type = virtual broadcast❺

create routing circuit tunnel_broad network protocol netware ipx❻
set routing circuit tunnel_broad network protocol netware ipx
   netware network number %x235f

create routing circuit tunnel_broad network protocol netware ipx❼
   adjacency tunnel_broad_adj1
set routing circuit tunnel_broad network protocol netware ipx
   adjacency tunnel_broad_adj1 encapsulation ip address 23.45.67.89❽

create routing circuit tunnel_broad network protocol netware ipx
   adjacency tunnel_broad_adj2
set routing circuit tunnel_broad network protocol netware ipx
   adjacency tunnel_broad_adj2 encapsulation ip address 34.56.78.91

enable routing circuit tunnel_broad❾
enable routing circuit tunnel_broad network protocol netware ipx
enable routing circuit tunnel_broad network protocol netware ipx
   adjacency tunnel_broad_adj1
enable routing circuit tunnel_broad network protocol netware ipx
   adjacency tunnel_broad_adj2
```

❶ Create the tunnel circuit, specifying that it is a point-to-point circuit.

❷ Create NETWORK PROTOCOL entities of types APPLETALK and NETWARE IPX for this circuit, since it will be used for both types of routing.

❸ Specify the destination IP address to be used in the IP header of the encapsulated packets.

❹ Enable the tunnel circuit and the NETWORK PROTOCOL entities.

❺ Create a broadcast tunnel circuit.

❻ Create the NETWORK PROTOCOL entity to allow it to route NetWare IPX packets.

❼ Create an adjacency for each destination (up to a maximum of five).

❽ Set the IP address of each destination.

❾ Enable the routing circuit, the NETWORK PROTOCOL entity, and the adjacency.

G.6 Configuring X.25 Gateway Functions

This example illustrates the NCL commands required to set up a DTE, and the server clients and filters required for X.25 gateway functions. The gateway created is open; that is, any Client system can use it to make calls, and any remote DTE can make calls to its server clients.

The example assumes that the following entities have been created and enabled:

X25 ACCESS
X25 SERVER
X25 PROTOCOL
SESSION CONTROL

The DTE will use the DEC WANcontroller 618 Network Interface Card used in Section G.3, so a DEVICE UNIT entity does not need to be set up here.

```
create modem connect line dte-line1 communication port W618-4-1, -
    profile  "NORMAL" ❶
set modem connect line dte-line1 speed 19200 ❷
set modem connect line dte-line1 modem control full ❸
set modem connect line dte-line1 suppress test indicator true ❹
enable modem connect line dte-line1

create lapb link dte-link1 profile  "PSS" ❺
set lapb link dte-link1 physical line modem connect line dte-line1 ❻
set lapb link dte-link1 acknowledge timer 1216 ❼
set lapb link dte-link1 holdback timer 608
set lapb link dte-link1 maximum data size 261 ❽
set lapb link dte-link1 window size 7 ❾
enable lapb link dte-link1

create x25 protocol dte dte-1 profile  "PSS" ❿
set x25 protocol dte dte-1 link service provider lapb link dte-link1 ⓫
set x25 protocol dte dte-1 inbound dte class PSS-class ⓬
set x25 protocol dte dte-1 x25 address 1234567890 ⓭
set x25 protocol dte dte-1 outgoing list  [[1..128]] ⓮
set x25 protocol dte dte-1 minimum packet size 16 ⓯
set x25 protocol dte dte-1 maximum packet size 128
set x25 protocol dte dte-1 default packet size 128
set x25 protocol dte dte-1 minimum window size 1
set x25 protocol dte dte-1 maximum window size 2
set x25 protocol dte dte-1 default window size 2
```

❶ Create a line on which to set up the DTE, and specify that it is to use port 1 of the Network Interface Card used in Section G.3; see Section 18.2.2.

❷ Set the line speed to 19.2 kbits/s.

❸ The line will take note of all the modem control signals; see Section 18.2.2.

❹ The Test Mode signal on the line will not be monitored.

❺ Create the LAPB data link, and specify the profile for the data link. Digital assigns a profile for each PSDN that it supports. Note that this profile must be the same as that used for the DTE; see Section 18.2.2.

❻ Tell the LAPB data link to use the line created above.

❼ See Section 18.2.4 (acknowledge timer) and Section 18.2.5 (holdback timer).

❽ Set the maximum frame size to the value supplied by the PSDN; see Section 18.2.7.

❾ Set the frame-level window size to the value supplied by the PSDN; see Section 18.2.6.

❿ Create the DTE, specifying that it is to use the same profile as the LAPB link; see Section 18.2.2.

⓫ Specify the data link that the DTE is to use; see Section 18.2.2.

⓬ Specify the DTE Class attribute that calls received on this DTE will acquire. For convenience, this is made the same as the DTE class to which the DTE belongs; see Section 18.3.

⓭ Set the DTE address, as specified by the PSDN; see Section 18.2.2.

⓮ Specify the channel numbers that this DTE will use for both-way and outgoing-only calls. By default, incoming calls will use channels 1 to 4095; see Section 18.2.2.

⓯ Specify the minimum, maximum and default packet sizes and window sizes that the DTE will use. Note that you only need to specify maximum and minimum values if the subscription options for the DTE include Flow Control Parameter Negotiation; see Section 18.2.10.2.

```
create x25 access dte class PSS-class type local ⓰
set x25 access dte class PSS-class local dtes (dte-1)
enable x25 protocol dte dte-1 ⓱

create x25 access filter filter-1 ⓲
set x25 access filter filter-1 priority 1
set x25 access filter filter-1 call data value %xfafa ⓳
set x25 access filter filter-1 call data mask %xffff

create x25 server client client-1 ⓴
set x25 server client client-1 filters (filter-1) ㉑
set x25 server client client-1  node made:.up.name ㉒
enable x25 server client client-1

create session control known tower made:.up.name towers  -
    {  -
     {  -
      {%X0113},  -
      {DNA$ProtID$SessCtlV3  Number = 25 },  -
      {dna$protid$nsp},  -
      {DNA$ProtID$RoutingV3  37:231499500156:00-a5:08-00-2b-19-0e-6c:20  }  -
    }    -
    } ㉓
```

⓰ Create a DTE class, and place the DTE in it. The DTE class is based on their profile name: all DTEs in a DTE class should connect to the same X.25 network; see Section 18.3.

⓱ Enable the DTE after putting it in its DTE class; this ensures that the DTE is available for use through the DTE class.

⓲ Create a filter to be used to select incoming calls for a particular group of Client systems; see Section 18.5.2.

⓳ This filter will match calls with a call data value of FAFA; see Section 18.5.4.

⓴ Now create a server client to represent a Client system to which incoming calls will be passed; see Section 18.4.2.

㉑ Specify that it will use the filter created above to select calls; see Section 18.4.5.

㉒ Specify the Client system represented by the server client; see Section 18.4.4.

㉓ Define the Client system in the Known Towers database; see Section 1.11.

```
create x25 access security filter default ㉔
set x25 access security filter default acl ((identifier =( * ), -
    access = ALL)) ㉕

create x25 access security dte class default ㉖
create x25 access security dte class default remote dte matchall -
    remote address prefix * ㉗
set x25 access security dte class default remote dte matchall -
    rights identifiers (PSI$OPEN_SECURITY) ㉘
set x25 access security dte class default remote dte matchall -
    acl ((identifier = ( * ), access = ALL)) ㉙

create x25 server security nodes secnodes-client-1 ㉚
set x25 server security nodes secnodes-client-1 nodes { *... }
set x25 server security nodes secnodes-client-1 -
    rights identifiers { PSI$OPEN_SECURITY }
```

㉔ By default, when a filter is created, the value of its SECURITY FILTER attribute is set to DEFAULT. You must now create this DEFAULT security filter; see Section 20.3.2. The filter FILTER-1 is now protected by this default security filter.

㉕ Because any remote DTE is to be allowed to access all filters, set the ACL so that any call (that is, one with any rights identifier) is granted ALL access. Any remote system can now access filter FILTER-1, and can therefore make calls to made:.up.name and another:.madeup.name; see Section 20.3.2.

㉖ By default, when a DTE class is created, the value of its SECURITY DTE CLASS attribute is set to DEFAULT. You must now create this security DTE class; see Section 20.3.2. Calls received on DTE class PSS-CLASS are now granted rights according to the REMOTE DTE entities in this default security DTE class.

㉗ Create a remote DTE, giving it a wildcard for an address prefix. This ensures that it will match a call from any remote DTE; see Section 20.3.2.

㉘ Grant the remote DTE a rights identifier; the value of the rights identifier does not matter. This rights identifier will now match the wildcard rights identifier in the ACL of the Default security filter. Therefore, any remote DTE can access FILTER-1; see Section 20.3.2.

㉙ Now set up outgoing security: ensure that any Client system can make calls to any remote DTE by granting ALL access to the wildcard rights identifier; see Section 20.5.2.

㉚ Create a SECURITY NODES entity to grant a rights identifier to all Client systems in the same namespace as the DECNIS; see Section 20.5.2. Note that it does not matter what rights identifier is granted to this SECURITY NODES entity.

G.7 Setting Up Security for X.25 Gateway Functions

This example illustrates the NCL commands required to set up security for the X.25 gateway functions created in Section G.6 so that only specified remote DTEs can make incoming calls, and only specified Client systems can make outgoing calls.

The example assumes that the system has been set up using the NCL commands in Section G.6, up to and including ㉒.

Security in this example is to be set up as follows:

Incoming Security

Remote DTE address 34561234577234 is allowed to make reverse-charge calls to made:.up.name and another:.madeup.name.

No calls of any kind are to be allowed from DTEs whose address begins with 2356.

Any other remote DTE can call made:.up.name and another:.madeup.name, provided the remote systems pay for the call.

Outgoing Security

made:.up.name and another:.madeup.name can call any remote DTE, provided the remote system pays for the call.

made:.up.name can call remote DTE 45671232345678, and pay for the call.

Any calls by other Client systems are disallowed.

```
set x25 access filter filter-1 security filter secfilt-1 ❶

create x25 access security filter secfilt-1 ❷
set x25 access security filter secfilt-1 acl ((identifier =(access_all), -
    access = ALL), -
    (identifier = (access_remote), access = REMOTE_CHARGE), -
    (identifier = (access_none), access = NONE))
```

❶ Specify that the filter for CLIENT-1 is to use security filter SECFILT-1 for incoming security, rather than DEFAULT.

❷ Create this security filter, and give it an access control list that specifies three rights identifiers, and the associated access actions.

```
create x25 access security dte class default remote dte can_access -
    remote address prefix 34561234577234 ❸
set x25 access security dte class default remote dte can_access -
    rights identifier (access_all)

create x25 access security dte class default remote dte cant_access -
    remote address prefix 2356 ❹
set x25 access security dte class default remote dte cant_access -
    rights identifier (access_none)

create x25 access security dte class default remote dte restrict_access -
    remote address prefix * ❺
set x25 access security dte class default remote dte restrict_access -
    rights identifier (access_remote)

create x25 server security nodes madeupname ❻
set x25 server security nodes madeupname nodes {made:.up.name}
set x25 server security nodes madeupname rights identifiers {made_sec}

create x25 server security nodes anothermadeupname ❼
set x25 server security nodes madeupname nodes {another:.madeup.name}
set x25 server security nodes madeupname rights identifiers {another_sec}

create x25 access security dte class default remote dte free -
    remote address prefix 45671232345678 ❽
set x25 access security dte class default remote dte free -
    acl ((identifier = (made_sec), access = ALL),-
    (identifier = (another_sec), access = REMOTE)) ❾
set x25 access security dte class default remote dte free -
    rights identifiers (access_remote) ❿

set x25 access security dte class default remote dte restrict_access -
    acl ((identifier = (made_sec), access = REMOTE_CHARGE), -
         (identifier = (another_sec), access = REMOTE_CHARGE)) ⓫
```

❸ Create a remote DTE for the DTE that is allowed to make reverse charge calls to both Client systems, and give it the rights identifier ACCESS_ALL. In this way, incoming calls from this remote DTE will acquire this rights identifier; the security filter SECFILT-1 specifies that calls with this rights identifier have REMOTE_CHARGE access.

❹ Create a remote DTE for the DTEs that are not allowed to make any incoming calls, and give it the rights identifier ACCESS_NONE. In this way, incoming calls from remote DTEs whose addresses begin with 2356 will acquire this rights identifier; the security filter SECFILT-1 specifies that calls with this rights identifier have access of NONE.

❺ Create a remote DTE for all other DTEs, and give it the rights identifier ACCESS_REMOTE. In this way, incoming calls from remote DTEs whose addresses do not match another remote DTE will acquire this rights identifier; the security filter SECFILT-1 specifies that calls with this rights identifier have access of REMOTE_CHARGE.

❻ Create a SECURITY NODES entity for made:.up.name, and assign a rights identifier.

❼ Create a SECURITY NODES entity for another:.madeup.name, and assign a rights identifier.

❽ Create a remote DTE for the DTE that made:.up.name can call (and pay for the call). Specify in the remote DTE's ACL that only Client systems having the rights identifier MADE_SEC can make nonreverse charge calls to this remote DTE.

❾ Include an ACE for identifier ANOTHER_SEC so that another:.madeup.name can call this remote DTE.

❿ Assign this rights identifier so that any incoming calls from this remote DTE will have remote charge access to filter FILTER-1.

⓫ A remote DTE with a wildcard address prefix already exists (RESTRICT_ACCESS, created above). Set an ACL for this remote DTE, specifying that Client systems having the rights identifier MADE_SEC and ANOTHER_SEC can make reverse charge calls to this remote DTE.

G.8 Configuring an X.25 DA Routing Circuit to Route Both DECnet/OSI and IP Traffic

This example illustrates the NCL commands required to set up an X.25 dynamically assigned routing circuit, and the associated manual routing entities.

The example assumes that the DTE set up in Section G.6 already exists on the DECNIS.

```
create routing circuit x25_da_circ1 type x25 da ❶
set routing circuit x25_da_circ1 data link entity (x25 access) ❷
set routing circuit x25_da_circ1 Manual L2Only Mode FALSE ❸
set routing circuit x25_da_circ1 recall timer 60 ❹
set routing circuit x25_da_circ1 idle timer 30 ❺
set routing circuit x25_da_circ1 maximum svc adjacencies 10 ❻
set routing circuit x25_da_circ1 manual data link sdu size 1492 ❼
set routing circuit x25_da_circ1 X25 filter { x25_da_osi_filt } ❽
set routing circuit x25_da_circ1 X25 filter { x25_da_ip_filt } ❾
set routing circuit x25_da_circ1 template x25_da_osi_temp ❿
set routing circuit x25_da_circ1 ip template x25_da_ip_temp ⓫
create x25 access template x25_da_osi_temp ⓬
set x25 access template x25_da_osi_temp call data %x81 ⓭
set x25 access template x25_da_osi_temp dte class pss-class ⓮

create x25 access template x25_da_ip_temp ⓯
set x25 access template x25_da_ip_temp call data %xcc ⓰
set x25 access template x25_da_ip_temp dte class pss-class ⓱
```

❶ Create the routing circuit, and specify its type; see Section 9.17.

❷ The data link for all X.25 circuits is the X25 ACCESS module; see Section 9.17.

❸ Allow the circuit to handle both Level 1 and Level 2 traffic; see Section 9.25.

❹ See Section 9.21.4.

❺ See Section 9.21.7.

❻ The number of SVC adjacencies is related to the number of SVCs allowed on this circuit; see Section 9.21.6.

❼ Specify the maximum size of data link message allowed on the circuit; see Section 9.21.2.

❽ Associate a filter with the circuit to select the OSI traffic for this circuit; see Section 9.16.

❾ Associate a filter with the circuit to select the IP traffic for this circuit; see Section 9.16.

❿ Associate a template with the circuit for outgoing OSI traffic on this circuit; see Section 9.14.

⓫ Associate a template with the circuit for outgoing IP traffic on this circuit; see Section 9.14.

⓬ Now create the templates associated with the circuit. This is the one for OSI traffic.

⓭ Call data for OSI traffic should be set to have 81 as the first octet; see Section 9.14.3.

⓮ Specify the DTE class that will be used to make calls on this circuit. This DTE class was created in Section G.6, and contains just one DTE.

⓯ Create the IP template.

⓰ Call data for IP traffic should have CC as the first octet; see Section 9.14.3.

⓱ Specify the DTE class that will be used to make calls on this circuit.

```
create x25 access filter x25_da_osi_filt ⓲
set x25 access filter x25_da_osi_filt call data mask %xff ⓳
set x25 access filter x25_da_osi_filt call data value %x81

create x25 access filter x25_da_ip_filt ⓴
set x25 access filter x25_da_ip_filt call data mask %xff ㉑
set x25 access filter x25_da_ip_filt call data value %xcc

enable routing circuit x25_da_circ1
create routing circuit x25_da_circ1 reachable address osi_domain2 -
    address prefix 37:2356: ㉒
set routing circuit x25_da_circ1 reachable address osi_domain2 -
    dte addresses { 78564353245778 }, mapping manual ㉓
set routing circuit x25_da_circ1 reachable address osi_domain2 cost 20 ㉔
enable routing circuit x25_da_circ1 reachable address osi_domain2

create routing circuit x25_da_circ1 ip reachable address ip_domain6 -
    destination {address= 67.78.0.0 ,mask = 255.255.0.0 } ㉕
set routing circuit x25_da_circ1 ip reachable address ip_domain6 -
    next hop 67.78.34.56 ㉖
set routing circuit x25_da_circ1 ip reachable address ip_domain6 -
    dte addresses { 45362684532194 } ㉗
set routing circuit x25_da_circ1 ip reachable address ip_domain6 -
    metric 20 ㉘
enable routing circuit x25_da_circ1 ip reachable address ip_domain6
```

⓲ Create the filter to be used to select calls carrying OSI traffic; see Section 18.5.2.

⓳ Set the call user data mask and value to check for a value of 81 in the first octet of call data.

⓴ Create the filter to be used to select calls carrying IP traffic; see Section 18.5.2.

㉑ Set the call user data mask and value to check for a value of CC in the first octet of call data.

㉒ All routing over X.25 DA circuits is manual. Therefore, you must set up reachable addresses for each routing domain you want to connect. Set up an OSI reachable address, to allow OSI traffic to be sent over this circuit to another OSI routing domain; see Section 16.16. Specify sufficient leading digits of the NSAP addresses in the foreign domain to uniquely identify them.

㉓ Specify the remote DTE address to be used to reach the foreign domain, and that the DTE address is not to be automatically derived from the target NSAP address; see Section 16.16.

㉔ Set the routing cost of using this reachable address.

㉕ Create a reachable address for IP traffic over this circuit, specifying the IP address and subnet mask of the IP network to be reached via this circuit; see Section 10.14.

㉖ Specify the IP address of the system to which this circuit will send IP packets; see Section 10.14.

㉗ Specify the remote DTE address to be used to reach the IP network.

㉘ Set the IP metric to be used when using this reachable address.

G.9 Configuring an X.25 Static Outgoing Circuit for IP (EGP) Routing

This example illustrates the NCL commands required to set up an X.25 Static Outgoing circuit, and configure it for IP routing using the EGP protocol. It also illustrates how to propagate IS–IS routes from one Autonomous System into another.

The circuit uses the DTE set up in Section G.6.

```
create routing circuit x25_so_circ2 type x25 static outgoing ❶
set routing circuit x25_so_circ2 data link entity (x25 access) ❷
set routing circuit x25_so_circ2 L1 cost 20
set routing circuit x25_so_circ2 recall timer 60
set routing circuit x25_so_circ2 manual data link sdu size 1492
set routing circuit x25_so_circ2 transmit verifier %x1234
set routing circuit x25_so_circ2 receive verifier %xfedc
set routing circuit x25_so_circ2 template x25_so_circ2 ❸
set routing circuit x25_so_circ2 subnet address {address= 12.23.34.45 -
    ,mask= 255.255.0.0 } ❹
enable routing circuit x25_so_circ2

create x25 access template x25_so_circ2 ❺
set x25 access template x25_so_circ2 destination dte address 9876543210 ❻
set x25 access template x25_so_circ2 call data %x81 ❼
set x25 access template x25_so_circ2 dte class pss-class ❽

create routing egp group egp1 ❾
set routing egp group egp1 autonomous system number 125 ❿
enable routing egp group egp1

create routing egp group egp1 egp neighbor egp1-neighbor -
    ip address 47.34.78.10, circuit x25_so_circ2 ⓫
enable routing egp group egp1 egp neighbor egp1-neighbor
```

❶ Create the routing circuit, and specify its type.

❷ Since it is an X.25 routing circuit, specify the data link as the X25 ACCESS module.

❸ Associate a template with the circuit. Note that in this case, only one template is required. For X25 DA circuits, two templates should be used (see Section G.8).

❹ Set the IP address and subnet mask for the circuit; see Section 10.5.

❺ Create the template associated with this circuit; see Section 9.14.

❻ You must specify the remote DTE to which the circuit will connect.

❼ Agree the call data to be used with the remote DTE.

❽ Specify the DTE class to be used for the circuit. This class was set up in Section G.6.

❾ Set up an EGP group to represent the autonomous system with which this routing circuit will connect; see Section 10.13.3.

❿ Specify the Autonomous System Number of this autonomous system; see Section 10.13.3. This is a number assigned by the Network Information Center at SRI International.

⓫ Create an EGP neighbor for the IP router in Autonomous System 125 with which this circuit will connect. Specify the IP address of the router; see Section 10.13.3. Specify the circuit to be used for connecting with Autonomous System 125.

```
create routing control protocol egp-1 type egp ⓬
set routing control protocol egp-1 autonomous system number 125 ⓭
enable routing control protocol egp-1

create routing control protocol egp-1 propagate destination - ⓮
    pass_all_l1 filter action pass
set routing control protocol egp-1 propagate destination - ⓯
    pass_all_l1 address match type prefix
set routing control protocol egp-1 propagate destination - ⓰
    pass_all_l1 sources {routing control protocol is-islevel1}
enable routing control protocol egp-1 propagate destination pass_all_l1

create routing control protocol egp-1 propagate destination - ⓱
    pass_all_l2 filter action pass, source route attributes -
    {isis internal}
set routing control protocol egp-1 propagate destination -
    pass_all_l2 address match type prefix
set routing control protocol egp-1 propagate destination -
    pass_all_l2 sources {routing control protocol is-islevel2}
enable routing control protocol egp-1 propagate destination pass_all_l2
```

⓬ Create a CONTROL PROTOCOL entity of type EGP. Routing automatically creates control protocols for IS-IS Level 1 and Level 2, called IS-ISLevel1 and IS-ISLevel2 respectively.

⓭ Associate the control protocol with the EGP group created earlier by specifying the autonomous system number of the EGP group.

⓮ Create a propagate destination for passing IS-IS Level 1 routes into EGP.

⓯ Specify the matching as Prefix. With prefix matching, when a domain is not specified, all Level 1 IS-IS routes will be propagated.

⓰ Specify the type of route that is to be passed to EGP.

⓱ Now create a propagate destination for passing IS-IS Level 2 routes into EGP.

G.10 Configuring CONS LAN/WAN Relay Functions

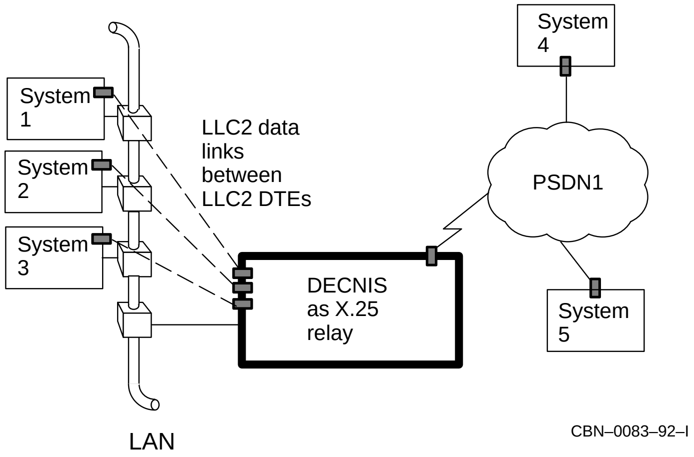
This example illustrates the NCL commands required to set up the LLC2 DTEs, and relay clients necessary to use the X.25 relay functions of the DECNIS.

In Figure G–1, the DECNIS is used to allow all the systems on the LAN to communicate among themselves, and also to allow communication between the systems on the LAN and Systems 4 and 5.

You must set up three LLC2 DTEs, one for each system on the LAN. In addition, you must set up a synchronous DTE, to communicate with PSDN1. The synchronous DTE is set up as described in Section G.6.

```
create llc2 sap sap1 ❶
set llc2 sap sap1 lan station csma-cd station csmacd_station1 ❷
set llc2 sap sap1 local lsap address 7e ❸
enable llc2 sap sap1

create llc2 sap sap1 link sys1-link ❹
set llc2 sap sap1 link sys1-link remote mac address 23-45-14-fe-9f-35 ❺
set llc2 sap sap1 link sys1-link remote lsap address 7e ❻
enable llc2 sap sap1 link sys1-link

create x25 protocol dte sys1-dte profile "ISO8881" ❼
set x25 protocol dte sys1-dte link service provider llc2 sap sap1
    link sys1-link ❽
```

Figure G–1 Using the DECNIS as a CONS LAN/WAN Relay



❶ First, create the three LLC2 DTEs for communicating with Systems 1, 2, and 3. The first step is to create a SAP. In this example, this SAP will be used for all three LLC2 DTEs on the DECNIS.

❷ This SAP will use the CSMA/CD station set up in Section G.1.

❸ Use the default local LSAP address for LLC2 calls.

❹ Create the data link to be used by the first DTE, for communicating with System 1.

❺ Specify the hardware address of System 1.

❻ Set the remote LSAP address to the value that System 1 uses for its local LSAP address.

❼ Create the DTE for communicating with System 1, and specify the LLC2 profile.

❽ Associate this DTE with the data link created earlier.

```
set x25 protocol dte sys1-dte x25 address 1234 ❾
set x25 protocol dte sys1-dte inbound dte class sys1-class

create x25 access dte class sys1-class type local ❿
add x25 access dte class sys1-class local dtes {sys1-dte}
enable x25 protocol dte sys1-dte

create llc2 sap sap1 link sys2-link ⓫
set llc2 sap sap1 link sys2-link remote mac address 34-56-25-0f-a0-46
set llc2 sap sap1 link sys2-link remote lsap address 7e
enable llc2 sap sap1 link sys2-link

create x25 protocol dte sys2-dte profile "ISO8881"
set x25 protocol dte sys2-dte link service provider llc2 sap sap1
    link sys2-link
set x25 protocol dte sys2-dte x25 address 2345
set x25 protocol dte sys2-dte inbound dte class sys2-class

create x25 access dte class sys2-class type local
add x25 access dte class sys2-class local dtes {sys2-dte}
enable x25 protocol dte sys2-dte

create x25 relay ⓬
enable x25 relay

create x25 access filter sys1-filter ⓭
set x25 access filter sys1-filter priority 1
set x25 access filter sys1-filter call data value %x3434
set x25 access filter sys1-filter call data mask %xffff

create x25 relay client sys1 ⓮
set x25 relay client sys1 filters {sys1-filter} ⓯
set x25 relay client sys1 dte class sys1-class ⓰
set x25 relay client sys1 rights identifiers {psi$open_security} ⓱
enable x25 relay client sys1
```

❾ Agree with System 1 on the DTE address to be used for this DTE. System 1 and the DECNIS could use the same DTE address at each end of the link.

❿ Put the DTE in a DTE class and enable it, to make it available through the DTE class. SYS1-DTE should be the only member of this DTE class.

⓫ Create the DTE for communicating with System 2 in exactly the same way as for System 1, except that this time there is no need to create the SAP. The commands for creating the DTE for communicating with System 3 are not shown.

Create the synchronous DTE as in Section G.6.

⓬ Ensure that the X25 RELAY entity exists.

⓭ Create the filter for the relay client, and set it to select calls that contain 3434 in the user data of the call request. Any system wanting to relay calls to System 1 must specify this call user data in the outgoing call.

⓮ Create a relay client for relaying calls to System 1.

⓯ Associate a filter with the relay client.

⓰ Specify the DTE class that will be used by the DECNIS for relaying calls to System 1.

⓱ Allocate a rights identifier to the relay client. In this example, X.25 security as in Section G.6 has been set up. The security filter used by filter Sys1-Filter is not specified, so it will use the security filter called Default. This allows all incoming calls to access the filter Sys1-Filter, and thus relay client Sys1.

Similarly, the security DTE class used by DTE class Sys1-Class is not specified, so it will use the security DTE class called Default. This allows calls bearing any rights identifier to access any remote DTE.

Now create relay clients for Systems 2 and 3, and one for PSDN1. Create reachable addresses if required.

G.11 Configuring Bridge Functions

This example illustrates the NCL commands required to set up the DECNIS as a bridge. It will use the CSMA/CD connection created in Section G.1 as one bridge port, another CSMA/CD connection (assumed to be already set up) as another, and the HDLC connection set up in Section G.3 as a remote bridge port.

The bridge will not be allowed to forward any Loopback protocol frames, and will only receive and forward frames with address 20-F2-39-30-30-AE on two of its three bridge ports. In addition, one of its bridge ports will not be allowed to forward any PID format frames.

```
create bridge
set bridge lb100 spanning tree compatibility ieee 802.1d ❶
set bridge root priority 255 ❷
enable bridge

create bridge port bridge-1 port number 1 ❸
set bridge port bridge-1 data link entity csma-cd station csmacd_station1 ❹
set bridge port bridge-1 port cost 64 ❺
enable bridge port bridge-1

create bridge port bridge-2 port number 2
set bridge port bridge-2 data link entity csma-cd station csmacd_station2
set bridge port bridge-2 port cost 64
enable bridge port bridge-2

create bridge port bridge-3 port number 3, type remote
set bridge port bridge-3 data link entity hdlc link hdlc_link1 logical -
    station hdlc_stat1 ❻
set bridge port bridge-3 port cost 200
set bridge port bridge-3 other pid disposition filter ❼
enable bridge port bridge-3

create bridge filter type 90-00 ❽
set bridge filter type 90-00 port set {} ❾

create bridge static physical address 20-f2-39-30-30-ae ❿
block bridge static physical address 20-f2-39-30-30-ae port set {bridge-3} ⓫
```

❶ Force the bridge to use the IEEE 802.1d implementation of the spanning tree algorithm; see Section 21.9.3.

❷ Set the root priority. The value depends on the other bridges in your extended LAN, and whether you want the bridge to be the root bridge; see Section 21.12.

❸ Create a bridge port, specifying a name and number; see Section 21.5.3.

❹ Specify the data link it is to use; in this case, it is to use the CSMA/CD data link set up in Section G.1.

❺ Specify the port cost. This will determine whether the bridge is placed in backup mode, or is the designated bridge for a LAN segment; see Section 21.13.

❻ Specify that this bridge port is to use the HDLC data link set up in Section G.3. This will therefore be a remote bridge port.

❼ Prevent any PID format frames from being received or forwarded on this bridge port; see Section 22.9.2

❽ Create a FILTER TYPE for the Loopback Ethernet protocol; the protocol format for Ethernet frames is *XX-XX*; the value for Loopback is 90-00.

❾ Specify which bridge ports are allowed to receive and forward frames with this protocol. By setting the PORT SET characteristic to null, no bridge ports may receive or forward such frames; see Section 22.14.4.

❿ Create a STATIC PHYSICAL ADDRESS, to restrict bridging of frames bearing this address either as source or destination; see Section 22.10.1.

⓫ Use the BLOCK command to prevent a bridge port from receiving or forwarding frames bearing this address; see Section 22.14.3.

G.12 Creating an ATM Permanent Circuit for DECnet/OSI and IP Routing

This example illustrates the NCL commands required to create a ATM Permanent circuit.

```
create multiplexed interface ❶

create atm connection management

create atm multiprotocol encapsulation

create device unit W631-7 name W631-7 ❷
enable device unit W631-7

create multiplexed interface line w631-7-0 comm port W631-7-0, -
    interface type  ds3 , -
    framing type  ds3-cbitparity ❸

set multiplexed interface line W631-7-0 -
    clock source network, -
    line length 130 ❹

create multiplexed interface line W631-7-0 logical channel W631-7-0 -
       map {0} ❺

create atm connection management line W631-7-0
set atm connection management line W631-7-0 -
    lower layer entity multiplexed interface line W631-7-0 -
    logical channel W631-7-0, -
    plcp disabled, -
    cell scrambling disabled ❻

create atm connection management line W631-7-0 pvc W631-7-0 -
    virtual circuit identifier 32, -
    virtual path identifier 0, -
    aal type aal5, -
    encapsulation type llc encapsulation ❼
```

❶ Create the Multiplexed Interface, ATM Connection Management, and ATM Multiprotocol Encapsulation modules.

❷ Create and enable a device unit for this Network Interface Card; see Section 8.4. The ATM line will use an ATMcontroller 631 Network Interface Card in slots 6 and 7 of the DECNIS 600 backplane.

❸ Create the Multiplexed Interface line. The ‘communication port name’ must map to the ‘operational communication port’ of the device unit entity. Set up the interface type and framing type as specified by the service provider, for example, DS3 (for T3) and DS3-CBitParity.

❹ Set the Multiplexed Interface line clock timing source as the ‘network’. Measure and set the line length to ensure that the necessary signal boosting is used. For example, 130 feet.

❺ Create the Multiplexed Interface logical channel map by specifying ‘map {0}’. This makes the entire bandwidth available for the installed device.

❻ Create the ATM Connection Management line. Note that there is always a one-to-one map of the ATM Connection Management line to the Multiplexed Interface line logical channel. Disable/enable cell scrambling and PLCP framing as required.

❼ Create an ATM Connection Management PVC. Note that there can be multiple PVCs on each ATM Connection Management line. The VCI valid range is 32 through 1023. The VPI valid range is 0 through 255 with 0 as the default. Specify that the ATM Adaptation Layer is of type AAL5 and that the ATM line uses LLC encapsulation.

```
set atm connection management line W631-7-0 pvc W631-7-0 -
    peak rate 100, -
    minimum guaranteed rate 100, -
    traffic shaping priority 1 ❽

create atm multiprotocol encapsulation link W631-7-0
set atm multiprotocol encapsulation link W631-7-0 lower layer entity -
    atm connection management line W631-7-0 pvc W631-7-0 ❾

set atm multiprotocol encapsulation link W631-7-0 maximum pdu size 4490 ❿

create routing circuit W631-7-0 type atm permanent ⓫
set routing circuit W631-7-0 -
   data link entity atm multiprotocol encapsulation link W631-7-0 ⓬
set routing circuit W631-7-0 network protocols {ip,iso8473}, -
   manual data link sdu size 4490 ⓭
set routing circuit W631-7-0 -
   subnet address {address=1.2.3.200, mask=255.255.255.0} ⓮

enable multiplexed interface line W631-7-0 logical channel W631-7-0 ⓯

enable multiplexed interface line W631-7-0

enable atm connection management line W631-7-0

enable atm multiprotocol encapsulation link W631-7-0

enable routing circuit W631-7-0
```

❽ Set the values for the virtual circuit rates and priority. For a single virtual circuit, all rates will be 100% and the priority is 1 (i.e. the highest).

❾ Create the ATM Multiprotocol Encapsulation Link and associate it with the the ATM Connection Management PVC.

❿ Set the maximum PDU size.

⓫ Create an ATM Permanent routing circuit.

⓬ Associate the routing circuit with the ATM Multiprotocol Encapsulation Link.

⓭ Set the routing network protocols. Set the maximum size of the largest data link message to be sent over the circuit to 4490.

⓮ Set the IP address and subnet mask for the IP interface.

⓯ Enable the ATM entities.

Index