DEC Network Integration Server

Management

Order Number: TBS

Revision/Update Information: This is a revised manual

Software Version: DECNIS™ T2.3

First Printing, March 1992
Revised, March 1994

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.

© Digital Equipment Corporation 1992, 1994.

All Rights Reserved.

The following are trademarks of Digital Equipment Corporation: Bookreader, DDCMP, DEC, DECbrouter 90, DECmcc, DECnet, DECNIS, DECrouter, DECUS, DECvoice, DECwindows, DNA, MASSBUS, MicroVAX, OpenVMS, Packetnet, Pathworks, PDP, POLYCENTER, Q–bus, Q22–bus, RSX, ULTRIX, 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.

NetWare is a registered trademark of Novell, Inc.

OSF and OSF/1 are registered trademarks of the Open Software Foundation, Inc.

TransLAN and Vitalink are registered trademarks of Vitalink Communications Corporation.

UNIX is a registered trademark of Unix System Laboratories, Inc., a wholly-owned subsidiary of Novell, Inc.

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

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

Preface xxxvii

Part I Managing the System

1 Tools for Managing the DECNIS

1.1 Introduction: SNMP and CMIP 1–1
1.1.1 SNMP (Simple Network Management Protocol) 1–1
1.1.2 DEC CMIP (Common Management Information Protocol) . . . 1–1
1.2 Management Model: CMIP 1–2
1.2.1 Management Modules 1–2
1.2.2 Entities 1–2
1.2.3 Entity Attributes 1–2
1.2.4 Example 1–2
1.2.5 Entities and Management 1–3
1.2.6 Entities Implemented in the DECNIS 1–3
1.3 Using NCL 1–5
1.3.1 What Is NCL? 1–5
1.3.2 Requirement: Install DECNIS Software on Management Hosts 1–5
1.3.3 Starting NCL and Issuing Commands 1–5
1.3.4 Security for NCL Commands 1–5
1.3.5 Example NCL Command 1–5
1.3.6 Using a Default Node 1–6
1.3.7 Setting Default Security for NCL Commands 1–6
1.4 Static and Dynamic Management 1–6
1.4.1 Dynamic Management: Definition 1–6
1.4.2 Static Management: Definition 1–6
1.5 NCL Script Files and CMIP Files 1–6
1.5.1 NCL Script Files 1–6
1.5.2 CMIP Files 1–6
1.5.3 Converting an NCL Script File to a CMIP File 1–7

1.6 Using the DECNIS Configurator to Create the NCL Script File 1–7
1.6.1 DECNIS Configurator Description 1–7
1.6.2 Modifying the Permanent Configuration Using the Configurator 1–7
1.7 User NCL Script Files 1–8
1.7.1 Introduction 1–8
1.7.2 Types of User NCL Script File 1–9
1.7.3 How the DECNIS Uses the User NCL Script Files 1–9
1.7.4 Example 1–11
1.8 DECdns 1–11
1.8.1 Introduction 1–11
1.8.2 Local Naming Option (LNO) 1–11
1.9 Translation of Node Names on the DECNIS 1–12
1.9.1 Introduction 1–12
1.9.2 Tower Sets 1–12
1.10 Entering Tower Sets in the Known Towers Database 1–12
1.10.1 Introduction 1–12
1.10.2 Entering Tower Sets in the Known Towers Database Using the Configurator 1–12
1.10.3 Entering Tower Sets in the Known Towers Database Using NCL Commands 1–13
1.10.4 Tower Structure 1–13
1.10.5 Example: Adding a Tower Set Using NCL 1–14
1.11 Creating MOP Circuits for Loopback Testing 1–14
1.11.1 Introduction 1–14
1.11.2 Procedure 1–15
1.12 DECmcc 1–16

2 Configuring SNMP on the DECNIS

2.1 Introduction 2–1
2.2 MIBs 2–1
2.2.1 MIB Definition 2–1
2.2.2 Groups and Objects 2–1
2.3 Overview 2–2
2.4 Access Control 2–3
2.4.1 Community Names 2–3
2.4.2 Example 2–3
2.5 MIBs Supported by the DECNIS 2–3
2.5.1 Location of MIBs 2–4
2.6 Traps 2–5
2.6.1 Definition 2–5

2.6.2 Traps Supported by the DECNIS 2–5
2.7 Configuring the DECNIS to Be Manageable Using SNMP 2–5
2.7.1 Introduction 2–5
2.7.2 Configuring the DECNIS to Respond to SNMP Requests 2–6
2.7.3 Example 2–7
2.7.3.1 Result 2–8
2.7.4 Using NCL to Configure the DECNIS to Send Traps 2–8
2.7.5 Using SNMP to Configure the DECNIS to Send Traps 2–9
2.8 Using SNMP to Manage the DECNIS 2–10

3 Setting Up Event Logging on the DECNIS

3.1 Event Logging on the DECNIS 3–1
3.1.1 Outbound Event Streams 3–1
3.1.2 Example 3–1
3.2 Event Sinks 3–1
3.2.1 Requirement to Set Up Event Sinks 3–1
3.2.2 Structure of Tower Set for an Event Sink 3–2
3.2.3 Example 3–3
3.2.4 Using DECmcc as the Event Sink 3–3
3.3 Setting Up Event Logging 3–3
3.3.1 Introduction 3–3
3.3.2 Procedure 3–4
3.4 Disconnecting the DECNIS from the Event Sink 3–5
3.4.1 Introduction 3–5
3.4.2 Breaking the Connection Immediately 3–5
3.4.3 Breaking the Connection in an Orderly Manner 3–5
3.4.4 Reestablishing the Connection 3–5
3.5 Disabling Event Streams 3–5
3.5.1 Introduction 3–5
3.5.2 Procedure 3–6
3.6 Connection Timers 3–6
3.6.1 Introduction 3–6
3.6.2 Connect Retry Timer 3–6
3.6.2.1 Changing the Connect Retry Timer 3–7
3.6.3 Disconnect Timer 3–7
3.6.3.1 Setting the Disconnect Timer 3–7
3.7 Event Filtering on the DECNIS 3–7
3.7.1 Introduction 3–7
3.7.2 Specific Filters 3–7
3.7.3 Global Filters 3–8
3.7.4 Catchall Filter 3–8
3.7.5 Operation of Event Filters 3–8

3.7.6 Recommendation 3–8
3.8 Events Blocked by Default 3–8
3.8.1 Overriding Default Blocking 3–10
3.9 Setting Up Specific Event Filters 3–10
3.9.1 Introduction 3–10
3.9.2 To Block Events 3–11
3.9.3 Example: Blocking 3–11
3.9.4 To Pass Events 3–11
3.9.5 Example: Passing 3–11
3.10 Setting Up Global Event Filters 3–12
3.10.1 Introduction 3–12
3.10.2 To Block Events 3–12
3.10.3 Example: Blocking 3–12
3.10.4 To Pass Events 3–12
3.10.5 Example: Passing 3–13
3.11 Setting Up a Catchall Event Filter 3–13
3.11.1 Introduction 3–13
3.11.2 To Block Events 3–13
3.11.3 To Pass Events 3–13
3.12 Removing Specific and Global Event Filters 3–14
3.12.1 Introduction 3–14
3.12.2 Removing Specific Event Filters 3–14
3.12.3 Removing Global Event Filters 3–14
3.13 Testing Event Streams and Filters 3–15
3.13.1 Introduction: TESTEVENT Command 3–15
3.13.2 Command Structure 3–15
3.13.3 Example 3–15

4 System-Level Management

4.1 Introduction 4–1
4.2 Security for NCL 4–1
4.2.1 Need for Security 4–1
4.2.2 NCL Security Mechanism 4–1
4.2.3 Changing the Username and Password for NCL 4–1
4.2.4 Example 4–2
4.3 Security for the Common Trace Facility 4–2
4.3.1 Need for Security 4–2
4.3.2 CTF Security Mechanism 4–2
4.3.3 Changing the Username and Password for CTF 4–2
4.3.4 Example 4–3
4.4 Preventing Unauthorized Loading 4–3
4.4.1 Need for Security 4–3

4.4.2 Setting a MOP Verification Value 4–3
4.4.3 Do Not Include Commands in NCL Script File 4–3
4.5 Assigning a Name to the DECNIS 4–4
4.6 Security for SNMP 4–4
4.6.1 Security for SNMP Requests 4–4
4.6.2 Security for Traps 4–4
4.7 Setting the System Time on the DECNIS 4–4
4.8 MIB-II: System Group 4–5

5 Connections to the Network

5.1 Introduction 5–1
5.1.1 NCL Commands 5–1
5.2 Use of Network Connections by the DECNIS 5–1
5.3 Port Naming Scheme 5–1
5.3.1 Port Name Format 5–1
5.3.2 Examples 5–2
5.4 Adding a Network Interface Card 5–2
5.4.1 Inserting the Network Interface Card 5–2
5.4.2 Configuring the New Network Interface Card 5–2
5.4.3 W622 and L602 LED Displays 5–3
5.5 Live Insertion 5–4
5.5.1 Definition 5–4
5.5.2 Self-Test 5–4
5.5.3 Removing the Network Interface Card 5–4
5.5.4 How a Card's Microcode Is Loaded 5–4
5.5.5 Restriction: F621 Network Interface Cards 5–4
5.5.6 Examples 5–4
5.6 Creating a CSMA/CD Connection 5–5
5.6.1 Introduction 5–5
5.6.2 Procedure 5–5
5.7 Creating an FDDI Connection 5–5
5.7.1 Introduction 5–5
5.7.2 Procedure 5–6
5.7.3 Enabling the Ring Purger on an FDDI Connection 5–6
5.8 Creating a VCP Connection 5–7
5.8.1 Introduction 5–7
5.8.2 Network Interface Card for VCP 5–7
5.8.3 Procedure 5–7
5.8.4 Routing Over VCP Data Links 5–8
5.8.5 Bridging Over VCP Data Links 5–8
5.9 Creating an HDLC Connection 5–8
5.9.1 Introduction 5–8

5.9.2 Procedure 5–9
5.10 Changing the Acknowledge and Holdback Timers on an HDLC Connection 5–10
5.10.1 Function of the Acknowledge Timer 5–10
5.10.2 Function of the Holdback Timer 5–10
5.10.3 Guidelines for Setting Timers 5–10
5.10.4 Setting the Acknowledge Timer 5–11
5.10.5 Setting the Holdback Timer 5–11
5.11 Changing Preferred Window Size on an HDLC Connection 5–11
5.11.1 Function of Preferred Window Size 5–11
5.11.2 Guideline for Setting Preferred Window Size 5–11
5.11.3 Setting the Preferred Window Size 5–11
5.12 HDLC Transmit Window Size 5–12
5.13 HDLC and Frame Forwarding Services 5–12
5.14 Creating a DDCMP Connection 5–12
5.14.1 Introduction 5–12
5.14.2 Procedure 5–13
5.15 Changing the Retransmit and Holdback Timers on a DDCMP Connection 5–14
5.15.1 Function of the Retransmit Timer 5–14
5.15.2 Function of the Holdback Timer 5–14
5.15.3 Guidelines for Setting Timers 5–15
5.15.4 Setting the Retransmit Timer 5–15
5.15.5 Setting the Holdback Timer 5–15
5.16 Changing the Transmit Window on a DDCMP Connection 5–15
5.16.1 Function of Transmit Window 5–15
5.16.2 Guideline for Setting Transmit Window Size 5–16
5.16.3 Setting the Transmit Window Size 5–16
5.17 Creating a PPP Connection 5–16
5.17.1 Introduction 5–16
5.17.2 Procedure 5–16
5.17.3 DECnet Phase IV Routing Over PPP: Interoperability with Other Vendors 5–17
5.18 Creating a CHDLC Connection 5–18
5.18.1 Introduction 5–18
5.18.2 Procedure 5–18
5.19 Creating a PPP or CHDLC Connection Using Frame Relay 5–19
5.19.1 Introduction 5–19
5.19.2 How PPP and CHDLC Use Frame Relay 5–19
5.19.3 Defining a PVC for the Connection 5–20
5.19.4 Function of the Management Protocol 5–20
5.19.5 Restrictions 5–20
5.19.6 Procedure 5–21

5.20 Creating a PPP Connection to a DEC WANrouter/Frame Relay 100/500 5–24
5.20.1 Restrictions 5–24
5.20.2 PPP Network Control Protocols 5–24
5.20.3 Procedure 5–24
5.21 Deleting a Connection 5–25
5.21.1 Reason for Deleting 5–25
5.21.2 Disabling Connections 5–25
5.21.3 Before Deleting a Connection 5–25
5.21.4 Deleting a CSMA/CD Connection 5–25
5.21.5 Deleting a VCP Connection 5–25
5.21.6 Deleting an HDLC Connection 5–26
5.21.7 Deleting a DDCMP Connection 5–26
5.21.8 Deleting a PPP Connection 5–27
5.21.9 Deleting a PPP Connection Over Frame Relay 5–27
5.22 Identifying Management Entities from the Hardware Port 5–28
5.22.1 Introduction 5–28
5.22.2 CSMA-CD STATION Name 5–28
5.22.3 FDDI STATION Name 5–28
5.22.4 MODEM CONNECT LINE Name 5–29
5.22.5 HDLC LINK Name 5–29
5.22.6 HDLC LINK LOGICAL STATION Name 5–29
5.22.7 DDCMP LINK Name 5–29
5.22.8 DDCMP LOGICAL STATION Name 5–29
5.22.9 FRBS CHANNEL Name 5–30
5.22.10 FRBS CHANNEL CONNECTION Name 5–30
5.22.11 PPP LINK Name 5–30
5.23 Using SNMP to Manage Connections 5–31

Part II Managing Routing

6 Managing Routing: General

6.1 Introduction 6–1
6.1.1 NCL Commands 6–1
6.2 Entities Associated with a Routing Circuit 6–2
6.3 Adding a CSMA/CD Routing Circuit 6–2
6.3.1 Introduction 6–2
6.3.2 Procedure 6–2
6.4 Adding an FDDI Routing Circuit 6–2
6.4.1 Introduction 6–2
6.4.2 Procedure 6–6

6.5 Adding an HDLC or DDCMP Routing Circuit 6–6
6.5.1 Introduction 6–6
6.5.2 Procedure 6–6
6.6 PPP Routing Circuits 6–7
6.6.1 Introduction 6–7
6.6.2 Defining Protocols for PPP Circuits 6–7
6.6.3 Default Protocols 6–7
6.6.4 PPP Circuits and DECnet Phase IV 6–7
6.6.5 System Resources 6–8
6.7 Adding a PPP Routing Circuit 6–8
6.7.1 Introduction 6–8
6.7.2 Procedure 6–8
6.8 CHDLC Routing Circuits 6–9
6.8.1 Introduction 6–9
6.9 Adding a CHDLC Routing Circuit 6–9
6.9.1 Introduction 6–9
6.9.2 Procedure 6–9
6.10 Changing the Phase IV Routing Vector Timer on PPP and CHDLC Circuits 6–10
6.10.1 Introduction 6–10
6.10.2 Procedure 6–10
6.11 X.25 Routing Circuits 6–11
6.11.1 Introduction 6–11
6.11.2 Types of X.25 Routing Circuit 6–11
6.12 Templates for X.25 Routing Circuits 6–12
6.12.1 Template Function 6–12
6.12.2 Which X.25 Routing Circuits Require Templates 6–12
6.12.3 Template Characteristics 6–12
6.13 Creating and Deleting X.25 Templates 6–13
6.13.1 Introduction 6–13
6.13.2 Procedure for Creating Templates 6–13
6.13.3 Procedure for Deleting Templates 6–14
6.14 Filters for X.25 Routing Circuits 6–14
6.14.1 Introduction 6–14
6.14.2 Which X.25 Routing Circuits Require Filters 6–14
6.14.3 Filter Characteristics 6–14
6.15 Adding an X.25 Dynamically Assigned Routing Circuit 6–15
6.15.1 Introduction 6–15
6.15.2 Routing Level 6–15
6.15.3 Procedure 6–15
6.16 Adding an X.25 Static Outgoing Routing Circuit 6–16
6.16.1 Introduction 6–16
6.16.2 Procedure 6–16

6.17 Adding an X.25 Static Incoming Routing Circuit 6–17
6.17.1 Introduction 6–17
6.17.2 Procedure 6–17
6.18 Adding an X.25 Permanent Routing Circuit 6–18
6.18.1 Introduction 6–18
6.18.2 Procedure 6–18
6.19 Changing Routing Circuit Characteristics 6–18
6.19.1 Introduction 6–18
6.19.2 Changing the Maximum Data Link Message Size 6–19
6.19.2.1 Description 6–19
6.19.2.2 Procedure 6–19
6.19.2.3 FDDI Routing Circuits 6–19
6.19.3 Changing the Originating Queue Limit 6–19
6.19.3.1 Description 6–19
6.19.3.2 Procedure 6–20
6.19.4 Changing the Recall Timer 6–20
6.19.4.1 Description 6–20
6.19.4.2 Procedure 6–20
6.19.5 Changing the Maximum Call Attempts 6–20
6.19.5.1 Description 6–21
6.19.5.2 Procedure 6–21
6.19.6 Changing the Maximum SVC Adjacencies 6–21
6.19.6.1 Description 6–21
6.19.6.2 Procedure 6–21
6.19.7 Changing the Idle Timer 6–21
6.19.7.1 Description 6–21
6.19.7.2 Procedure 6–21
6.19.8 Changing the Initial Minimum Timer 6–22
6.19.8.1 Description 6–22
6.19.8.2 Procedure 6–22
6.20 Using NCL to Monitor a Routing Circuit 6–22
6.21 Deleting a Routing Circuit 6–22
6.21.1 Introduction 6–22
6.21.2 Deleting Non-X.25 Routing Circuits 6–22
6.21.3 Deleting X.25 Routing Circuits 6–23
6.22 Changing the Routing Level of the DECNIS 6–23
6.22.1 Description of Routing Level 6–23
6.22.2 Cannot Change Routing Level of a Running System 6–24
6.22.3 Procedure 6–24
6.23 Preventing Routing Circuits from Routing Level 1 Traffic 6–24
6.23.1 Introduction 6–24
6.23.2 Procedure 6–24
6.23.3 Consequences of Changing from TRUE to FALSE 6–25

6.24 Routing Security 6–25
6.24.1 Introduction 6–25
6.24.2 Routing Security Mechanism 6–25
6.24.3 Circuit and System Routing Security 6–25
6.24.4 How Routing Security Works 6–25
6.25 Setting Up Circuit-Level Routing Security 6–27
6.25.1 Introduction 6–27
6.25.2 Procedure 6–27
6.26 Setting Up System-Level Routing Security 6–27
6.26.1 Introduction 6–27
6.26.2 Procedure 6–28
6.27 Configuring the DECNIS to Send Verifiers 6–29
6.27.1 Introduction 6–29
6.27.2 Procedure 6–29
6.28 Optimizing the Routing Database Sizes on the DECNIS 6–29
6.28.1 Introduction 6–29
6.28.2 How Routing Database Size Is Implemented on the DECNIS 6–29
6.28.3 OSPF Routing Characteristics 6–32
6.28.3.1 Connectivity: A Measure of the "Connectedness" of a Network 6–33
6.28.4 Dynamically Allocated DECNIS Memory 6–34
6.28.5 When to Change the Database Sizes from Their Defaults . . . 6–35
6.28.5.1 Allow for Later Expansion 6–35
6.28.6 How to Change the Routing Database Sizes 6–35
6.28.6.1 Use the Configurator 6–35
6.28.6.2 How the DECNIS Configurator Manages Routing Database Sizes 6–36
6.28.7 Choosing New Values for the Routing Databases 6–36
6.28.7.1 Total Memory Available 6–36
6.28.7.2 Choosing Which Database to Decrease 6–36
6.28.7.3 Example 6–37
6.28.8 Memory Allocation on the DECNIS 6–37

7 IP Routing

7.1 Introduction 7–1
7.1.1 NCL Commands 7–1
7.2 Using the DECNIS as an IP Router 7–2
7.2.1 Routing Circuits 7–2
7.2.2 Example IP Network 7–2
7.2.3 Forwarding IP Packets to Non-Digital Systems 7–2
7.2.4 Selecting an IP Protocol 7–4

7.3 IP Protocols 7–4
7.3.1 Protocols Supported by DECNIS 7–4
7.3.2 Autonomous Systems 7–4
7.4 IP Addressing 7–4
7.4.1 IP Address Format 7–4
7.4.2 Subnet Masks 7–5
7.4.3 Example 7–5
7.4.4 Further Reading 7–5
7.5 Assigning IP Addresses and Subnet Masks 7–5
7.5.1 Subnets 7–5
7.5.2 Multiple IP Addresses on a LAN Interface 7–5
7.5.3 LAN Routing Circuits 7–6
7.5.4 Point-to-Point Routing Circuits 7–6
7.5.5 PPP Routing Circuits 7–6
7.5.6 Procedure for Assigning IP Subnet Addresses 7–7
7.6 Default IP Address 7–7
7.6.1 Source Address in IP Packets 7–7
7.6.2 Setting a Default IP Address 7–8
7.7 Using the Integrated IS–IS Protocol for IP Routing 7–8
7.7.1 What Is IS–IS? 7–8
7.7.2 Routing Algorithm 7–8
7.7.3 Procedure 7–9
7.8 Configuring the DECNIS to Use the OSPF Protocol 7–9
7.8.1 Introduction to OSPF 7–9
7.8.2 OSPF Organization 7–9
7.8.3 Authentication 7–10
7.8.4 OSPF Interface Types 7–10
7.8.5 Router Classification 7–11
7.8.6 OSPF CONTROL PROTOCOL Entities 7–13
7.8.6.1 Using the DECNIS Configurator to Set Up OSPF 7–13
7.8.6.2 Creating OSPF AREA CONTROL PROTOCOL Entities 7–14
7.8.6.3 Example 1: Setting Up a Minimum OSPF Configuration 7–14
7.8.6.4 Example 2: Setting Up an OSPF Multiple Areas Configuration 7–16
7.8.6.5 Example 3: Connecting the OSPF Backbone Using Virtual Links 7–19
7.8.6.6 Example 4: Procedure for Authentication 7–20
7.8.7 Designated Routers 7–21
7.8.8 Using NBMA on Ethernet 7–21
7.8.9 Alternate Subnetworks 7–22
7.8.10 Unused Interfaces 7–22

7.9 Configuring the DECNIS to Use the RIP Protocol 7–22
7.9.1 RIP Instances 7–22
7.9.2 RIP Over Point-to-Point Circuits 7–23
7.9.3 Procedure for Setting Up RIP 7–23
7.9.4 Using the Configurator to Set Up RIP 7–27
7.10 The EGP Protocol 7–28
7.10.1 Example of Using the EGP Protocol 7–28
7.10.2 Routing Level Required to Run EGP 7–28
7.11 Configuring the DECNIS to Use the EGP Protocol 7–28
7.11.1 Example 7–28
7.11.2 Procedure 7–28
7.11.3 EGP Route Propagation and Filtering 7–32
7.11.4 Using the Configurator to Set Up EGP Propagation 7–32
7.12 Static Routes 7–33
7.12.1 Introduction 7–33
7.12.2 Example 7–33
7.12.3 Setting Up a Static Route 7–35
7.13 Default Routes 7–36
7.13.1 Definition 7–36
7.13.2 Types of Default Route 7–36
7.13.3 Creating a Static Default Route 7–37
7.14 Abbreviating IP Routes in IS-IS Level 2 Routers 7–38
7.14.1 Introduction 7–38
7.14.2 Example 7–38
7.14.3 Summary IP Information and Routing Level 7–38
7.14.4 Setting Up Summary IS-IS Addresses on the DECNIS 7–38
7.15 Setting Up Summary OSPF Addresses on the DECNIS 7–40
7.15.1 Introduction 7–40
7.16 Setting Up Summary Routes for an OSPF Area 7–40
7.16.1 Example 7–40
7.17 Using All Subnets Broadcast 7–41
7.17.1 Introduction 7–41
7.17.2 Example 7–42
7.17.3 Data Link Broadcast 7–42
7.17.4 Procedure 7–42
7.18 Using Proxy ARP 7–42
7.18.1 ARP (Address Resolution Protocol) 7–42
7.18.2 Proxy ARP: Definition 7–43
7.18.3 How Proxy ARP Operates on the DECNIS 7–43
7.18.4 Example 7–43
7.18.5 Setting Up Proxy ARP on the DECNIS 7–44
7.19 Using Alternative Subnet Addresses 7–44
7.19.1 Introduction 7–44

7.19.2 Example 7–45
7.19.3 Setting Up Alternative Subnet Addresses on the DECNIS . . 7–45
7.20 Using SNMP to Manage IP Routing 7–46

8 IP Route Propagation and Filtering

8.1 Introduction 8–1
8.1.1 NCL Commands 8–1
8.2 Introduction to Route Propagation and Filtering 8–1
8.3 Entities Controlling Route Propagation and Filtering 8–2
8.4 Route Filtering 8–5
8.4.1 What Happens Without Route Filtering? 8–5
8.4.1.1 Advertisement Information Received 8–5
8.4.1.2 Routing Tables 8–7
8.4.1.3 Routes Passed from the Routing Tables to the Forwarding Table 8–10
8.4.1.4 Filtering Received Routes 8–11
8.4.2 Setting Up Route Filtering 8–12
8.4.2.1 The RECEIVE DESTINATION Filter 8–12
8.4.2.2 Setting Up a RECEIVE DESTINATION Filter to Block Routes 8–13
8.4.2.3 Setting Up a Filter to Change a Protocol Preference 8–16
8.4.2.4 Disappearing Routes 8–18
8.4.2.5 Special Note About Filtering IS–IS and OSPF Routes . . . 8–19
8.4.2.6 Filtering All Routes of a Control Protocol 8–20
8.4.2.7 Filtering the Default Route 8–20
8.5 Route Propagation 8–20
8.5.1 What Happens Without Route Propagation? 8–20
8.5.1.1 Route Advertising 8–20
8.5.1.2 Advertising Routes Learnt by Another Routing Protocol 8–20
8.5.2 Setting Up Route Propagation 8–22
8.5.2.1 The PROPAGATE DESTINATION Filter 8–22
8.5.3 Setting Up a Filter to Propagate Routes 8–25
8.5.4 Advertising a Default Route 8–28
8.5.5 Advertising All Routes of a SOURCE Control Protocol 8–28
8.5.6 IS–IS Propagation Exceptions 8–29
8.5.7 OSPF Propagation Exceptions 8–29
8.6 Using the Configurator to Set Up Route Propagation 8–29
8.7 Examples of Setting Up Route Filtering and Propagation Using RIP, EGP, and IS–IS 8–31
8.8 Examples of Setting Up Route Propagation Using RIP and OSPF 8–34

8.8.1 An Example of Propagating RIP Routes into OSPF 8–34
8.8.2 An Example of Propagating OSPF Routes into RIP 8–35
8.9 IP Domains 8–36
8.9.1 Setting Up IP Domains 8–36
8.9.2 IP Domains and the ADDRESS MATCH TYPE 8–37
8.9.2.1 Overlapping IP Domains 8–38

9 DECnet/OSI Routing

9.1 Introduction 9–1
9.1.1 NCL Commands 9–1
9.2 DECNIS Addressing 9–1
9.2.1 Introduction 9–1
9.2.2 DECNIS Addresses and DECnet Phase IV Nodes 9–2
9.2.3 Address Structure 9–2
9.3 Deriving the Phase IV-Compatible NSAP Address 9–2
9.3.1 Introduction 9–2
9.3.2 Procedure 9–2
9.4 Changing the Phase IV Prefix 9–3
9.4.1 Restrictions 9–3
9.4.2 Procedure 9–3
9.5 Changing the Phase IV Address 9–4
9.5.1 Reboot the DECNIS 9–4
9.5.2 Procedure 9–4
9.6 Deriving the Phase V-Style NSAP Address 9–4
9.6.1 Introduction 9–4
9.6.2 Procedure 9–4
9.7 Changing the Phase V Area Address 9–5
9.7.1 Introduction 9–5
9.7.2 Procedure 9–5
9.8 NSAP Formats 9–5
9.8.1 Introduction 9–5
9.8.2 OSI Format 9–5
9.8.3 HRPF Format 9–5
9.9 Deciding Which Routing Algorithms to Run 9–6
9.9.1 Introduction 9–6
9.9.2 Procedure 9–6
9.9.3 Phase IV Cluster Aliases 9–6
9.10 Changing the Routing Algorithms 9–7
9.10.1 Consequences of Changing the Routing Algorithms 9–7
9.10.2 Reboot the DECNIS 9–7
9.10.3 Procedure 9–7
9.11 Changing the Circuit Cost 9–7

9.11.1 Introduction 9–7
9.11.2 Restrictions on Circuit Costs 9–7
9.11.3 Procedure 9–8
9.12 Circuit Priority and Designated Routers 9–8
9.12.1 Designated Router Definition 9–8
9.12.2 Designated Routers and Routing Level 9–8
9.12.3 Function of Designated Level 1 Router 9–8
9.12.4 Function of Designated Level 2 Router 9–8
9.13 Changing the Circuit Priority 9–9
9.13.1 Introduction 9–9
9.13.2 Procedure 9–9
9.14 Manual Routing 9–9
9.14.1 Definition 9–9
9.14.2 When to Use Manual Routing 9–9
9.15 Connecting to Foreign Routing Domains 9–10
9.15.1 What Is a Foreign Routing Domain? 9–10
9.15.2 Configuration Restriction 9–10
9.15.3 Procedure 9–10
9.16 Connecting to Routers Running a Different Routing Algorithm 9–12
9.16.1 Introduction 9–12
9.16.2 Definition: Interphase Link 9–12
9.16.3 Outbound Reachable Addresses 9–12
9.16.4 Inbound Reachable Addresses 9–12
9.16.5 Restriction: Phase IV Area Numbers 9–12
9.16.6 Example 9–12
9.16.7 Setting Up an Interphase Link 9–13
9.16.8 Using Interphase Links to Connect to All Possible Phase IV Areas 9–15
9.17 Connecting to LAN Systems Without Automatic Configuration 9–15
9.17.1 Introduction 9–15
9.17.2 Connecting to End Systems in the Same Area as the DECNIS 9–16
9.17.2.1 Example 9–16
9.17.3 Connecting to End Systems in a Different Area from the DECNIS 9–16

10 AppleTalk and IPX Routing

10.1 AppleTalk Routing 10–1
10.1.1 Introduction 10–1
10.1.2 AppleTalk Protocols Supported 10–1
10.1.3 License Required 10–2
10.1.4 Native and Tunnel Routing Circuits 10–2
10.1.5 Modes of AppleTalk Routing 10–2
10.1.6 AppleTalk Addresses 10–2
10.1.6.1 AppleTalk Addresses and Tunnel Circuits 10–3
10.1.7 AppleTalk Name 10–3
10.1.8 Setting Up AppleTalk Routing on a LAN Routing Circuit . . . 10–3
10.1.8.1 FDDI Restriction 10–3
10.1.8.2 Procedure 10–3
10.1.8.3 Disabling AppleTalk Routing Circuits 10–4
10.1.9 Seed Routers 10–4
10.1.9.1 Networks Without a Seed Router 10–5
10.1.10 AppleTalk Zones 10–5
10.1.10.1 Setting a Default AppleTalk Zone 10–6
10.1.10.2 AppleTalk Zone List 10–6
10.1.11 AppleTalk Characters 10–7
10.2 NetWare Routing 10–7
10.2.1 Introduction 10–7
10.2.2 NetWare Protocols Supported 10–7
10.2.3 License Required 10–7
10.2.4 Native and Tunnel Routing Circuits 10–8
10.2.5 NetWare Addresses 10–8
10.2.6 Setting Up NetWare IPX Routing on a LAN Routing Circuit 10–8
10.2.6.1 FDDI Restriction 10–8
10.2.6.2 Procedure 10–9
10.2.6.3 Disabling NetWare IPX Routing Circuits 10–9
10.2.7 Frame Types on LAN Circuits 10–10
10.2.7.1 Changing the Frame Type on a Routing Circuit 10–10
10.2.8 Periodic Routing Protocol Timer 10–10
10.2.8.1 Changing the Periodic Routing Protocol Timer 10–11
10.2.9 NetBIOS Broadcast 10–11
10.2.9.1 Turning NetBIOS Broadcast On and Off 10–11
10.3 Tunnel Circuits 10–11
10.3.1 Introduction 10–11
10.3.2 Maximum Number of Tunnel Circuits 10–12
10.3.3 Types of Tunnel Circuit 10–12
10.3.4 Source Address for Tunnel Circuits 10–12

10.3.5 Setting Up a Point-to-Point Tunnel Circuit for AppleTalk or NetWare IPX Routing 10–12
10.3.6 Setting Up a Broadcast Tunnel Circuit for NetWare IPX Routing 10–13

Part III Managing X.25

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

11.1 Introduction 11–1
11.1.1 General 11–1
11.1.2 Outgoing Calls 11–2
11.1.3 Incoming Calls 11–2
11.2 Managing DTEs 11–2
11.2.1 Introduction 11–2
11.2.2 Adding a DTE 11–2
11.2.3 Disable Loading/Dumping Over X.25 Lines 11–5
11.2.4 Changing the Acknowledge Timer 11–6
11.2.5 Changing the Holdback Timer 11–6
11.2.6 Changing the Window Size 11–7
11.2.7 Changing the Maximum Frame Size 11–7
11.2.8 Monitoring a DTE 11–7
11.2.8.1 X.25 Accounting 11–8
11.2.9 Deleting a DTE 11–8
11.2.10 Changing DTE Characteristics 11–8
11.2.10.1 Point-to-Point Links: DTE Interface Type 11–9
11.2.10.2 Packet and Window Size 11–9
11.3 Managing DTE Classes 11–11
11.3.1 Managing DTE Classes 11–11
11.3.2 Adding a DTE Class 11–11
11.3.3 Deleting a DTE Class 11–11
11.3.4 Changing Members of a DTE Class 11–12
11.4 Managing Server Clients 11–12
11.4.1 Introduction 11–12
11.4.2 Adding a Server Client 11–13
11.4.3 Deleting a Server Client 11–13
11.4.4 Changing the Client System Associated with a Server Client 11–14
11.4.5 Changing the List of Filters Associated with a Server Client 11–14
11.5 Managing Filters 11–14
11.5.1 Introduction 11–14

11.5.2 Adding a Filter 11–15
11.5.3 Deleting a Filter 11–15
11.5.4 Changing Filter Characteristics 11–17
11.6 Managing PVCs 11–17
11.6.1 Introduction 11–17
11.6.2 Adding a PVC 11–17
11.6.3 Deleting a PVC 11–18
11.7 Managing Groups 11–18
11.7.1 Introduction 11–18
11.7.2 Adding a Group 11–18
11.7.3 Deleting a Group 11–19
11.7.4 Changing Members of a Group 11–19

12 X.25 Relay

12.1 Description of the X.25 Relay 12–1
12.2 How the DECNIS Operates as an X.25 Relay 12–4
12.2.1 Introduction 12–4
12.2.2 Switching Calls Across a DECnet/OSI Network 12–6
12.2.3 Using X.25 Reachable Addresses to Translate NSAP Addresses 12–6
12.3 Managing Relay Clients 12–7
12.3.1 Adding a Relay Client 12–8
12.3.2 Deleting a Relay Client 12–9
12.3.3 Changing the List of Filters Associated with a Relay Client 12–9
12.4 Managing Remote DTE Classes 12–10
12.4.1 Introduction 12–10
12.4.2 Adding a Remote DTE Class 12–10
12.4.3 Deleting a Remote DTE Class 12–10
12.4.4 Changing the System Associated with a Remote DTE Class 12–10
12.5 Creating a Permanent Association Between Two DTEs 12–11
12.6 Managing Relay PVCs 12–13
12.6.1 Introduction 12–13
12.6.2 Adding a Relay PVC 12–13
12.6.3 Deleting a Relay PVC 12–14
12.7 LLC2 Communication and the CONS LAN/WAN Relay 12–14
12.7.1 Introduction 12–14
12.7.2 Operating the DECNIS as a CONS LAN/WAN Relay 12–14
12.8 Managing LLC2 DTEs 12–15
12.8.1 Introduction 12–15
12.8.2 Setting Up a LAN Connection to Use for LLC2 DTEs 12–16

12.8.3 Adding an LLC2 DTE 12–17
12.8.4 Deleting an LLC2 DTE 12–18
12.8.5 Changing LLC2 DTE Characteristics 12–19
12.9 Managing X.25 Reachable Addresses 12–19
12.9.1 Introduction 12–19
12.9.2 Adding an X.25 Reachable Address 12–19
12.9.3 Deleting an X.25 Reachable Address 12–20

13 X.25 Security

13.1 How Incoming Security Works 13–1
13.2 Setting Up Incoming Security 13–4
13.2.1 Introduction 13–4
13.2.2 Controlling Access to a Filter by a Remote DTE 13–4
13.2.3 Preventing Access to a Filter by a Remote DTE 13–6
13.2.4 Creating a Closed, Semiclosed, or Open System 13–6
13.2.4.1 Procedure 13–6
13.2.5 Applying the Same Incoming Security to All Filters on a System 13–7
13.2.6 Applying the Same Incoming Security to All Local DTE Classes 13–8
13.3 How Outgoing Security Works 13–8
13.4 Setting Up Outgoing Security 13–11
13.4.1 Introduction 13–11
13.4.2 Controlling Access to Remote DTEs by a Client System 13–11
13.4.3 Controlling Access to PVCs by a Client System 13–13
13.4.4 Preventing Access to a Remote DTE by a Client System 13–13
13.4.5 Preventing a Client System from Using the DECNIS as a Connector System 13–14
13.4.6 Creating a Closed, Semiclosed, or Open System 13–14
13.4.6.1 Procedure 13–15
13.4.7 Allowing Client Systems Outgoing Access to All Remote DTEs 13–15
13.4.8 Applying the Same Outgoing Security to All Local DTE Classes 13–16
13.5 X.25 Relay Security 13–16
13.5.1 How X.25 Relay Security Works 13–16
13.5.2 X.25 Relay Security and PVCs 13–19
13.6 ACLs 13–21
13.6.1 ACL Structure 13–21
13.6.2 Example ACL 13–21
13.6.3 Order of Checking ACEs 13–21

Part IV Managing Bridging

14 Setting Up the DECNIS as a Bridge

14.1 Introduction 14–1
14.1.1 NCL Commands 14–1
14.2 Bridging 14–1
14.2.1 Definition 14–1
14.2.2 Example 14–1
14.2.3 Local Bridging 14–2
14.2.4 Remote Bridging 14–3
14.2.4.1 Connecting to Vitalink TransLAN Bridges: VCP Bridging 14–3
14.2.5 Extended LANs: Definition 14–3
14.2.6 Further Reading 14–3
14.3 Entities Associated with a Bridge Port 14–3
14.4 Routing and Bridging of Protocols 14–6
14.4.1 Protocols that Will Not Be Bridged 14–6
14.4.2 Protocols that Can Be Bridged or Routed 14–6
14.4.3 How the DECNIS Decides How to Treat These Protocols 14–6
14.4.4 Switching Between Routing and Bridging 14–6
14.4.5 Implications for SNMP 14–6
14.4.6 Protocol Filtering 14–6
14.5 Setting Up the DECNIS as a Bridge 14–7
14.5.1 Introduction 14–7
14.5.2 DECNIS Requirements 14–7
14.5.3 Procedure 14–7
14.6 Using VCP Bridging 14–9
14.7 Using NCL to Monitor a Bridge Port 14–9
14.8 Deleting a Bridge Port 14–9
14.8.1 Introduction 14–9
14.8.2 Procedure 14–10
14.9 The Spanning Tree Algorithm 14–10
14.9.1 Definition 14–10
14.9.2 Bridge Hellos 14–10
14.9.3 Different Modes of the Spanning Tree Algorithm 14–11
14.10 Changing the Spanning Tree Algorithm on the DECNIS 14–11
14.10.1 Introduction 14–11
14.10.2 Switching Between Spanning Tree Algorithm Modes 14–11
14.10.3 Disabling the Spanning Tree Algorithm 14–12
14.11 Root Bridge and Designated Bridges 14–12
14.11.1 Root Bridge 14–12
14.11.2 Designated Bridges 14–12

14.11.3 Example 14–12
14.12 Changing Bridge Priority 14–14
14.12.1 Introduction 14–14
14.12.2 Procedure 14–14
14.13 Changing the Port Cost 14–14
14.13.1 Introduction 14–14
14.13.2 Procedure 14–14
14.14 Minimum-Sized Ethernet/802.3 Frame Compression 14–14
14.14.1 Definition 14–14
14.14.2 When to Use Frame Compression 14–15
14.14.3 How Frame Compression Works 14–15
14.14.4 Procedure: PPP Bridge Ports 14–15
14.14.5 Procedure: VCP Bridge Ports 14–16
14.15 Bridging and DECnet Phase IV Addresses 14–16
14.15.1 External Bridges and DECnet Phase IV Addresses 14–16
14.15.2 Example 14–16
14.15.3 Reason 14–16
14.15.4 Solutions 14–16
14.16 Remote Bridging and Loopback 14–18
14.17 Using SNMP to Manage Bridging 14–18

15 Bridge Filtering on the DECNIS

15.1 Introduction 15–1
15.1.1 General 15–1
15.1.2 NCL Commands 15–1
15.1.3 Entering Bridge Filtering Commands in the User NCL Script Files 15–1
15.2 Bridge Filtering 15–2
15.2.1 Definition 15–2
15.2.2 Filtering by Address 15–2
15.2.3 Filtering by Protocol Type 15–2
15.3 Frame and Protocol Types 15–3
15.3.1 Frame Types 15–3
15.3.2 Protocol Types 15–3
15.3.3 Examples of Protocol Types 15–3
15.3.4 Lists of Protocol Types 15–3
15.4 DECNIS Filtering Process 15–4
15.4.1 Introduction 15–4
15.4.2 Flowchart: Bridge Filtering 15–4
15.5 Management Entities Used for Protocol Filtering 15–4
15.5.1 Introduction 15–4
15.5.2 Filtering Entities 15–4

15.6 Permitting a Port to Bridge Only Specified Addresses 15–6
15.6.1 Introduction 15–6
15.6.2 Procedure 15–6
15.7 Permitting a Port to Bridge Only Specified Ethernet Protocol Types 15–7
15.7.1 Introduction 15–7
15.7.2 Procedure 15–7
15.8 Permitting a Port to Bridge Only Specified IEEE 802.2 Normal Format Protocol Types 15–8
15.8.1 Introduction 15–8
15.8.2 Procedure 15–8
15.9 Permitting a Port to Bridge Only Specified 802.2 SNAP SAP Protocol Types 15–9
15.9.1 Introduction 15–9
15.9.2 Procedure 15–9
15.10 Restricting Bridging of Certain Addresses to Specific Bridge Ports 15–10
15.10.1 Introduction 15–10
15.10.2 Procedure 15–10
15.11 Restricting Bridging of Certain Ethernet Protocols to Specific Bridge Ports 15–11
15.11.1 Introduction 15–11
15.11.2 Procedure 15–11
15.12 Restricting Bridging of Certain IEEE 802.2 Normal Format Protocols to Specific Bridge Ports 15–12
15.12.1 Introduction 15–12
15.12.2 Procedure 15–12
15.13 Restricting Bridging of Certain SNAP SAP Protocols to Specific Bridge Ports 15–13
15.13.1 Introduction 15–13
15.13.2 Procedure 15–13
15.14 Specifying Ports for Filtering 15–14
15.14.1 Introduction 15–14
15.14.2 PORT SET Characteristic 15–14
15.14.3 Use of the BLOCK and SET Command 15–14
15.14.4 Command Format 15–14
15.14.5 Requirement 15–15
15.14.6 Examples 15–15
15.14.7 Checking the Bridge Ports 15–16
15.15 Locking a Destination Address to a Specific Bridge Port 15–16
15.15.1 Introduction 15–16
15.15.2 Example 15–16
15.15.3 Procedure 15–16

15.16 Bridging of Null Internet Frames on FDDI Ports 15–18
15.16.1 Null Internet Frames on CSMA/CD Ports 15–18
15.17 Using SNMP to Manage Bridge Filtering 15–18

Part V Appendixes

A Management Modules Used by the DECNIS

A.1 Bridge Module A–1
A.2 CSMA-CD Module A–2
A.3 CHDLC Module A–2
A.4 DDCMP Module A–3
A.5 Device Module A–3
A.6 DTSS Clerk Module A–3
A.7 Event Dispatcher Module A–4
A.8 FDDI Module A–4
A.9 FRBS Module A–5
A.10 Hardware Module A–5
A.11 HDLC Module A–6
A.12 LAPB Module A–6
A.13 LLC2 Module A–7
A.14 Loopback Application Module A–7
A.15 Modem Connect Module A–7
A.16 MOP Module A–8
A.17 NSP Module A–8
A.18 PPP Module A–9
A.19 Routing Module A–10
A.20 Session Control Module A–11
A.21 SNMP Module A–11
A.22 X25 Access Module A–12
A.23 X25 Protocol Module A–12
A.24 X25 Relay Module A–13
A.25 X25 Server Module A–13

B Characteristic Attributes Supported by the DECNIS

B.1 Bridge Module B–2
B.1.1 BRIDGE Characteristics B–2
B.1.2 BRIDGE PORT Characteristics B–3
B.1.3 BRIDGE STATIC PHYSICAL ADDRESS Characteristics . . . B–3
B.1.4 BRIDGE STATIC MULTICAST ADDRESS Characteristics B–4

B.1.5 BRIDGE FILTER SAP Characteristics B–4
B.1.6 BRIDGE FILTER PID Characteristics B–4
B.1.7 BRIDGE FILTER TYPE Characteristics B–4
B.2 CHDLC Module B–5
B.2.1 CHDLC Characteristic B–5
B.2.2 CHDLC LINK Characteristics B–5
B.3 CSMA-CD Module B–6
B.3.1 CSMA-CD Characteristics B–6
B.3.2 CSMA-CD STATION Characteristics B–6
B.4 DDCMP Module B–7
B.4.1 DDCMP LINK Characteristics B–7
B.4.2 DDCMP LOGICAL STATION Characteristics B–7
B.5 Device Module B–8
B.5.1 DEVICE Characteristics B–8
B.5.2 DEVICE UNIT Characteristics B–8
B.6 Event Dispatcher Module B–9
B.6.1 EVENT DISPATCHER OUTBOUND STREAM Characteristics B–9
B.7 FDDI Module B–10
B.7.1 FDDI Characteristics B–10
B.7.2 FDDI STATION Characteristics B–10
B.7.3 FDDI STATION LINK Characteristics B–11
B.7.4 FDDI STATION PHY PORT Characteristics B–11
B.8 FRBS Module B–12
B.8.1 FRBS Characteristics B–12
B.8.2 FRBS CHANNEL Characteristics B–12
B.8.3 FRBS CHANNEL CONNECTION Characteristic B–13
B.9 Hardware Module B–13
B.9.1 HARDWARE Characteristics B–13
B.9.2 HARDWARE SLOT Characteristics B–14
B.10 HDLC Module B–15
B.10.1 HDLC Characteristic B–15
B.10.2 HDLC LINK Characteristics B–15
B.11 LAPB Module B–16
B.11.1 LAPB Characteristics B–16
B.11.2 LAPB LINK Characteristics B–16
B.12 LLC2 Module B–17
B.12.1 LLC2 Characteristic B–17
B.12.2 LLC2 SAP Characteristic B–17
B.12.3 LLC2 SAP Link Characteristics B–17
B.13 Loopback Application Module B–18
B.13.1 LOOPBACK APPLICATION Characteristics B–18
B.14 Modem Connect Module B–19

B.14.1 MODEM CONNECT Characteristic B–19
B.14.2 MODEM CONNECT LINE Characteristics B–19
B.15 MOP Module B–21
B.15.1 MOP Characteristics B–21
B.15.2 MOP CIRCUIT Characteristics B–21
B.15.3 MOP CLIENT Characteristics B–22
B.16 PPP Module B–23
B.16.1 PPP Characteristics B–23
B.16.2 PPP LINK Characteristics B–23
B.17 Routing Module B–25
B.17.1 ROUTING Characteristics B–25
B.17.2 ROUTING CONTROL PROTOCOL Characteristics B–28
B.17.3 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT Characteristics B–30
B.17.4 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT ADJACENCY Characteristics B–31
B.17.5 ROUTING CONTROL PROTOCOL PROPAGATE DESTINATION Characteristics B–32
B.17.6 ROUTING CONTROL PROTOCOL RECEIVE DESTINATION Characteristics B–33
B.17.7 ROUTING NETWORK PROTOCOL DOMAIN Characteristics B–33
B.17.8 ROUTING PERMITTED NEIGHBOR Characteristics B–34
B.17.9 ROUTING CIRCUIT Characteristics B–35
B.17.10 ROUTING CIRCUIT NETWORK PROTOCOL Characteristics B–38
B.17.11 ROUTING CIRCUIT NETWORK PROTOCOL ADJACENCY Characteristics B–39
B.17.12 ROUTING CIRCUIT REACHABLE ADDRESS Characteristics B–39
B.17.13 ROUTING EGP GROUP Characteristics B–40
B.17.14 ROUTING EGP GROUP EGP NEIGHBOR Characteristics B–40
B.17.15 ROUTING CIRCUIT IP REACHABLE ADDRESS Characteristics B–41
B.18 Session Control Module B–42
B.18.1 Session Control Characteristics B–42
B.18.2 SESSION CONTROL KNOWN TOWER Characteristics B–43
B.19 SNMP Module B–43
B.19.1 SNMP Characteristics B–43
B.19.2 SNMP COMMUNITY Characteristics B–43
B.20 X25 Access Module B–44
B.20.1 X25 ACCESS Characteristics B–44

B.20.2 X25 ACCESS DTE CLASS Characteristics B–44
B.20.3 X25 ACCESS FILTER Characteristics B–45
B.20.4 X25 ACCESS REACHABLE ADDRESS Characteristics B–46
B.20.5 X25 ACCESS SECURITY DTE CLASS REMOTE DTE Characteristics B–46
B.20.6 X25 ACCESS SECURITY FILTER Characteristics B–46
B.20.7 X25 ACCESS TEMPLATE Characteristics B–47
B.21 X25 Protocol Module B–48
B.21.1 X25 PROTOCOL Characteristic B–48
B.21.2 X25 PROTOCOL DTE Characteristics B–48
B.21.3 X25 PROTOCOL DTE PVC Characteristics B–50
B.21.4 X25 PROTOCOL GROUP Characteristics B–50
B.22 X25 Relay Module B–51
B.22.1 X25 RELAY CLIENT Characteristics B–51
B.22.2 X25 RELAY PVC Characteristics B–51
B.23 X25 Server Module B–52
B.23.1 X25 SERVER Characteristics B–52
B.23.2 X25 SERVER CLIENT Characteristics B–52
B.23.3 X25 SERVER SECURITY NODES Characteristics B–53

C NCL Commands to Set Up Event Sinks

C.1 Setting Up an OpenVMS System as an Event Sink C–1
C.2 Setting Up an ULTRIX or OSF/1 System as an Event Sink C–1

D Common Protocol Values and Formats

D.1 Ethernet Protocol Type Values D–1
D.1.1 Public Ethernet Protocol Types D–1
D.1.2 Digital Ethernet Protocol Types D–2
D.1.3 Publicly Available Private Ethernet Protocol Types D–3
D.2 IEEE 802.2 LSAP Protocol Values D–7
D.3 IEEE 802.2 SNAP Protocol IDs D–8

E Characteristic Values of the Default X25 ACCESS Template

F Connectivity Tables

F.1 Where to Start F–1
F.2 Checking Information Required for Connections to DECnet/OSI Systems F–2
F.2.1 Procedure F–2
F.3 Checking Information Required for Connections to DECnet Phase IV Systems F–7
F.3.1 Procedure F–7

G Management from DECnet–VAX Extensions Systems

G.1 Introduction G–1
G.2 Node Implementation Syntax on DECnet–VAX Extensions G–1
G.3 Tower Set Syntax on DECnet–VAX Extensions G–1
G.4 Logging Events to DECmcc on DECnet–VAX Extensions G–2
G.5 Testing Event Logging on DECnet–VAX Extensions G–2
G.6 NCL "with" Clause G–2
G.7 MOP Service Password when Loading from a DECnet–VAX Extensions System G–3

H Example NCL Scripts

H.1 Creating a CSMA/CD Routing Circuit for DECnet/OSI and IP (RIP) Routing H–2
H.2 Creating an FDDI Routing Circuit for DECnet/OSI, IP (IS–IS), AppleTalk, and NetWare IPX Routing H–4
H.3 Creating an HDLC Routing Circuit for DECnet/OSI and IP (IS–IS) Routing H–6
H.4 Creating a PPP Routing Circuit (Over Frame Relay) for DECnet/OSI and IP (IS–IS) Routing H–8
H.5 Creating Tunnel Circuits for AppleTalk and NetWare IPX Routing H–10
H.6 Configuring X.25 Gateway Functions H–12
H.7 Setting Up Security for X.25 Gateway Functions H–18
H.8 Configuring an X.25 DA Routing Circuit to Route Both DECnet/OSI and IP Traffic H–22
H.9 Configuring an X.25 Static Outgoing Circuit for IP (EGP) Routing H–28
H.10 Configuring CONS LAN/WAN Relay Functions H–32
H.11 Configuring Bridge Functions H–36

Index

Figures

1–1	Routing Module	1–2
1–2	Management Modules Implemented in the DECNIS	1–4
1–3	Managing the DECNIS Using NCL and the Configurator	1–10
2–1	Managing the DECNIS Using SNMP	2–2
3–1	Event Streams on the DECNIS	3–2
3–2	Operation of Event Filters in the Outbound Stream	3–9
6–1	Entities Required by LAN and VCP Routing Circuits	6–3
6–2	Entities Required by PPP and CHDLC Routing Circuits	6–4
6–3	Entities Required by X.25, HDLC, and DDCMP Routing Circuits	6–5
6–4	Security Checks Made by Routing	6–26
6–5	Connectivity	6–34
7–1	Example IP Configuration	7–3
7–2	Different Functions of an OSPF Router	7–12
7–3	OSPF Minimum Configuration	7–15
7–4	OSPF Multiple Area Configuration	7–17
7–5	Using a Virtual Link to Connect to the OSPF Backbone	7–20
7–6	Using RIP in an IP Network	7–24
7–7	Communication Among Autonomous Systems Using the EGP Protocol	7–29
7–8	Using EGP in an IP Network	7–30
7–9	Setting Up a Static Route	7–34
7–10	Using Abbreviated Routes	7–39
7–11	All Subnets Broadcast	7–41
7–12	DECNIS Using Proxy ARP	7–44
7–13	Routing Between Subnets on a LAN	7–45
8–1	Entities that Control Route Propagation and Filtering	8–3
8–2	Route Advertising	8–6
8–3	Multiple Occurrences of a Route	8–9
8–4	Routes Passed to the Forwarding Table	8–11
8–5	Protocol Instance ‘A’ - Route Filtering	8–14
8–6	IS–IS/OSPF Filtering	8–19
8–7	Route Propagation	8–21

8–8	Protocol Instance ‘A’—Route Advertising	8–24
8–9	Example IP Network Using Route Filtering and Propagation	8–31
8–10	IP Domains and Filters	8–39
9–1	Interphase Links	9–14
11–1	How Filters Handle Incoming Calls on the DECNIS	11–16
12–1	Principles of X.25 Relay	12–2
12–2	DECNIS Performing More Complex Relay Functions	12–3
12–3	DECNIS Switching Calls Locally	12–5
12–4	DECNIS Switching Calls Remotely	12–7
12–5	Switching Between PVCs Locally	12–11
12–6	Switching Between PVCs Remotely	12–12
12–7	DECNIS Performing as a CONS LAN/WAN Relay	12–15
13–1	How an Incoming Call Acquires Rights Identifiers	13–2
13–2	How X.25 Security Decides Whether to Permit an Incoming Call	13–3
13–3	How Outgoing Security Operates on the DECNIS	13–9
13–4	How Outgoing Security for PVCs Operates on the DECNIS	13–10
13–5	How Outgoing Security for X.25 Relay Functions Operates on the DECNIS	13–18
13–6	X.25 Security for Relay Functions	13–19
13–7	How Outgoing Security for X.25 Relay Functions on PVCs Operates on the DECNIS	13–20
14–1	DECNIS Operating as a Bridge	14–2
14–2	Entities Required by Local and Remote VCP Bridge Ports	14–4
14–3	Entities Required by Remote PPP, HDLC, and CHDLC Bridge Ports	14–5
14–4	Extended LAN Showing Path Costs	14–13
14–5	Illegal Routing and Bridging Between Two LANs	14–17
15–1	Bridge Filtering on the DECNIS	15–5
15–2	DECNIS Operating as a Bridge Between Two LANs	15–17
A–1	Bridge Module	A–1
A–2	CSMA-CD Module	A–2
A–3	CHDLC Module	A–2
A–4	DDCMP Module	A–3
A–5	Device Module	A–3
A–6	Event Dispatcher Module	A–4

A–7 FDDI Module A–4
A–8 FRBS Module A–5
A–9 Hardware Module A–5
A–10 HDLC Module A–6
A–11 LAPB Module A–6
A–12 LLC2 Module A–7
A–13 Modem Connect Module A–7
A–14 MOP Module A–8
A–15 NSP Module A–8
A–16 PPP Module A–9
A–17 Routing Module A–10
A–18 Session Control Module A–11
A–19 SNMP Module A–11
A–20 X25 Access Module A–12
A–21 X25 Protocol Module A–12
A–22 X25 Relay Module A–13
A–23 X25 Server Module A–13
H–1 Using the DECNIS as a CONS LAN/WAN Relay H–33

Tables

2–1 Traps Sent by the DECNIS 2–5
2–2 SNMP Traps: MIB Variables 2–9
3–1 DECNIS Events Blocked by Default 3–10
6–1 Types of X.25 Routing Circuit 6–11
6–2 Template Characteristics for X.25 Routing Circuits 6–12
6–3 ROUTING Attributes Relating to Routing Database Sizes . . . 6–30
6–4 OSPF General Control Protocol Characteristic Attributes . . 6–32
6–5 DECNIS Memory Required for Each Configuration Parameter 6–38
7–1 RIP Configurator Defaults 7–27
8–1 Entity Descriptions 8–4
8–2 Advertisement Information Received 8–7
8–3 Routing Table Data 8–8
8–4 RECEIVE DESTINATION Characteristics 8–13
8–5 Source Route Attributes 8–15
8–6 PROPAGATE DESTINATION Characteristics 8–23

8–7	ANNOUNCE ROUTE Attributes	8–27
8–8	ANNOUNCE METRIC Values	8–28
8–9	Address Matching	8–38
9–1	How to Choose the Routing Algorithms	9–6
10–1	Frame Types	10–10
15–1	Bridge Filtering Entities	15–6
B–1	BRIDGE Characteristics	B–2
B–2	BRIDGE PORT Characteristics	B–3
B–3	BRIDGE STATIC PHYSICAL ADDRESS Characteristics	B–3
B–4	BRIDGE STATIC MULTICAST ADDRESS Characteristics	B–4
B–5	BRIDGE FILTER SAP Characteristics	B–4
B–6	BRIDGE FILTER PID Characteristics	B–4
B–7	BRIDGE FILTER TYPE Characteristics	B–4
B–8	CHDLC Characteristics	B–5
B–9	CHDLC LINK Characteristics	B–5
B–10	CSMA-CD Characteristics	B–6
B–11	CSMA-CD STATION Characteristics	B–6
B–12	DDCMP LINK Characteristics	B–7
B–13	DDCMP LOGICAL STATION Characteristics	B–7
B–14	DEVICE Characteristic	B–8
B–15	DEVICE UNIT Characteristics	B–8
B–16	EVENT DISPATCHER OUTBOUND STREAM Characteristics	B–9
B–17	FDDI Characteristics	B–10
B–18	FDDI STATION Characteristics	B–10
B–19	FDDI STATION LINK Characteristics	B–11
B–20	FDDI STATION PHY PORT Characteristics	B–11
B–21	FRBS Characteristic	B–12
B–22	FRBS CHANNEL Characteristics	B–12
B–23	FRBS CHANNEL CONNECTION Characteristic	B–13
B–24	HARDWARE Characteristics	B–13
B–25	HARDWARE SLOT Characteristics	B–14
B–26	HDLC Characteristic	B–15
B–27	HDLC LINK Characteristics	B–15
B–28	LAPB Characteristic	B–16
B–29	LAPB LINK Characteristics	B–16

B–30 LLC2 Characteristic B–17
B–31 LLC2 SAP Characteristics B–17
B–32 LLC2 SAP LINK Characteristics B–17
B–33 LOOPBACK APPLICATION Characteristics B–18
B–34 MODEM CONNECT Characteristic B–19
B–35 MODEM CONNECT LINE Characteristics B–19
B–36 MOP Characteristics B–21
B–37 MOP CIRCUIT Characteristics B–21
B–38 MOP CLIENT Characteristics B–22
B–39 PPP Characteristics B–23
B–40 PPP LINK Characteristics B–23
B–41 ROUTING Characteristics B–25
B–42 ROUTING CONTROL PROTOCOL Characteristics B–28
B–43 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT Characteristics B–30
B–44 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT ADJACENCY Characteristics B–31
B–45 ROUTING CONTROL PROTOCOL PROPAGATE DESTINATION Characteristics B–32
B–46 ROUTING CONTROL PROTOCOL RECEIVE DESTINATION Characteristics B–33
B–47 ROUTING NETWORK PROTOCOL DOMAIN Characteristics B–33
B–48 ROUTING PERMITTED NEIGHBOR Characteristics B–34
B–49 ROUTING CIRCUIT Characteristics B–35
B–50 ROUTING CIRCUIT NETWORK PROTOCOL Characteristics B–38
B–51 ROUTING CIRCUIT NETWORK PROTOCOL ADJACENCY Characteristics B–39
B–52 ROUTING CIRCUIT REACHABLE ADDRESS Characteristics B–39
B–53 ROUTING EGP GROUP Characteristics B–40
B–54 ROUTING EGP GROUP EGP NEIGHBOR Characteristics B–40
B–55 ROUTING CIRCUIT IP REACHABLE ADDRESS Characteristics B–41
B–56 SESSION CONTROL Characteristics B–42
B–57 SESSION CONTROL KNOWN TOWER Characteristics B–43

B–58 SNMP Characteristic B–43
B–59 SNMP COMMUNITY Characteristics B–43
B–60 X25 ACCESS Characteristics B–44
B–61 X25 ACCESS DTE CLASS Characteristics B–44
B–62 X25 ACCESS FILTER Characteristics B–45
B–63 X25 ACCESS REACHABLE ADDRESS Characteristics B–46
B–64 X25 ACCESS SECURITY DTE CLASS REMOTE DTE Characteristics B–46
B–65 X25 ACCESS SECURITY FILTER Characteristics B–46
B–66 X25 ACCESS TEMPLATE Characteristics B–47
B–67 X25 PROTOCOL Characteristic B–48
B–68 X25 PROTOCOL DTE Characteristics B–48
B–69 X25 PROTOCOL DTE PVC Characteristics B–50
B–70 X25 PROTOCOL GROUP Characteristics B–50
B–71 X25 RELAY CLIENT Characteristics B–51
B–72 X25 RELAY PVC Characteristics B–51
B–73 X25 SERVER Characteristics B–52
B–74 X25 SERVER CLIENT Characteristics B–52
B–75 X25 SERVER SECURITY NODES Characteristics B–53
D–1 Public Ethernet Protocol Types D–1
D–2 Digital Ethernet Protocol Types D–2
D–3 Publicly Available Private Ethernet Protocol Types D–3
D–4 LSAP Protocol Values D–7
D–5 Digital IEEE 802.2 SNAP Protocol Identification Assignments D–8
E–1 Characteristic Values of ‘Default’ Template E–1
F–1 DECNIS Uses Link State at Level 1 and Link State at Level 2 . . . . . F–3
F–2 DECNIS Uses Routing Vector at Level 1 and Link State at Level 2 . . . . . F–4
F–3 DECNIS Uses Link State at Level 1 and Routing Vector at Level 2 . . . . . F–5
F–4 DECNIS Uses Routing Vector at Level 1 and Routing Vector at Level 2 . . . . . F–6
F–5 Information for Connecting to DECnet Phase IV Systems . . . F–7

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 host)
- ULTRIX™ (if using an ULTRIX load host)
- OSF/1® (if using an OSF/1 load host)
- IBM®-compatible Personal Computers running MS–DOS® (if using a PC load host)

Associated Documentation

Product Documentation

- *DEC Network Integration Server Introduction and Glossary*
- *DEC Network Integration Server Installation and Configuration*
- *DEC Network Integration Server Problem Solving*

  This is only available on line through the Bookreader™ or as a series of text files if you have a PC load host.
- *DEC Network Integration Server Event Messages* (supplied on line)

- *DEC Network Integration Server Release Notes* (supplied on line)

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

- *DECnet/OSI Routing Overview*

  This manual introduces routing on DECnet/OSI networks, and defines the basic routing terminology and concepts. It may be useful to read this manual before using the DECNIS.

- 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

Documentation Comments

If you have comments or suggestions for this manual or any of the DECNIS documents and you have access to the Internet, mail your comments electronically to the DECNIS writing group within Digital at the following address:

doc_quality@lkg.mts.dec.com

Alternatively, you can complete the postpaid Reader's Comments form at the end of this document.

Thank you.

Conventions

The following conventions are used in this manual:

Italics	This indicates variable information.
DECNIS	DEC Network Integration Server
decnis	This indicates that you should substitute the node name of the DECNIS. If you are using DECdns, enter the name registered in DECdns.
Prompts	The following prompts precede commands that you enter: For OpenVMS: `$` For ULTRIX: `#` 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 set up SNMP to manage the DECNIS.
- Chapter 3 explains how to set up event logging from the DECNIS.
- Chapter 4 describes system-level management.
- Chapter 5 describes how to set up synchronous and LAN connections.

1
Tools for Managing the DECNIS

1.1 Introduction: SNMP and CMIP

The DECNIS supports two management protocols:

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

1.1.1 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 2 describes how to configure the DECNIS to be manageable using SNMP.

1.1.2 DEC CMIP (Common Management Information Protocol)

DEC CMIP is a Digital-proprietary protocol. Using this protocol on a host system running MS–DOS, OpenVMS, ULTRIX, or OSF, you can monitor and change all the DECNIS functions, either dynamically or statically (see Section 1.4).

The remainder of this chapter describes the CMIP management model, and the tools that utilize CMIP.

1.2 Management Model: CMIP

1.2.1 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.

1.2.2 Entities

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

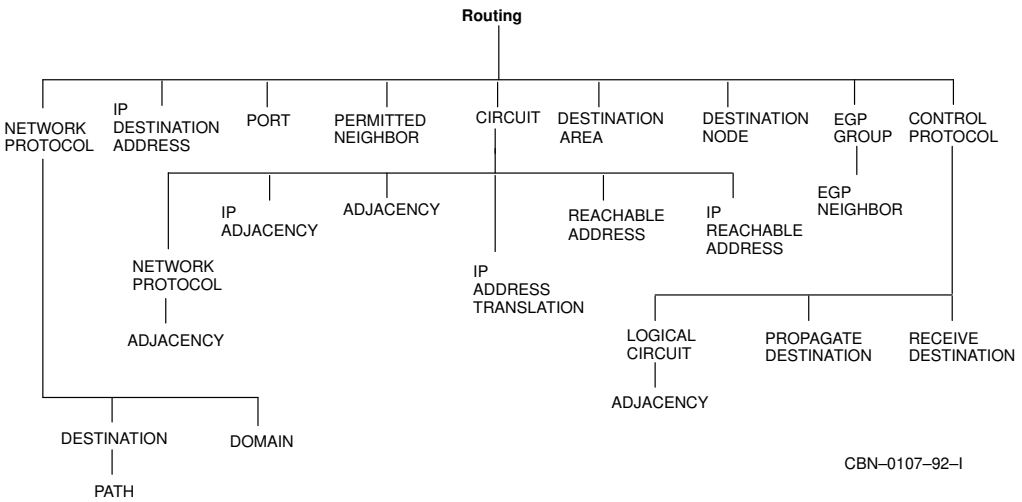
1.2.3 Entity Attributes

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

1.2.4 Example

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

Figure 1–1 Routing Module



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 characteristic 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.

1.2.5 Entities and Management

To manage the DECNIS, log on to a suitable host system and use the management tools described in the remainder of this chapter. In this way, you 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.
- Display the information for existing entities.

1.2.6 Entities Implemented in the DECNIS

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

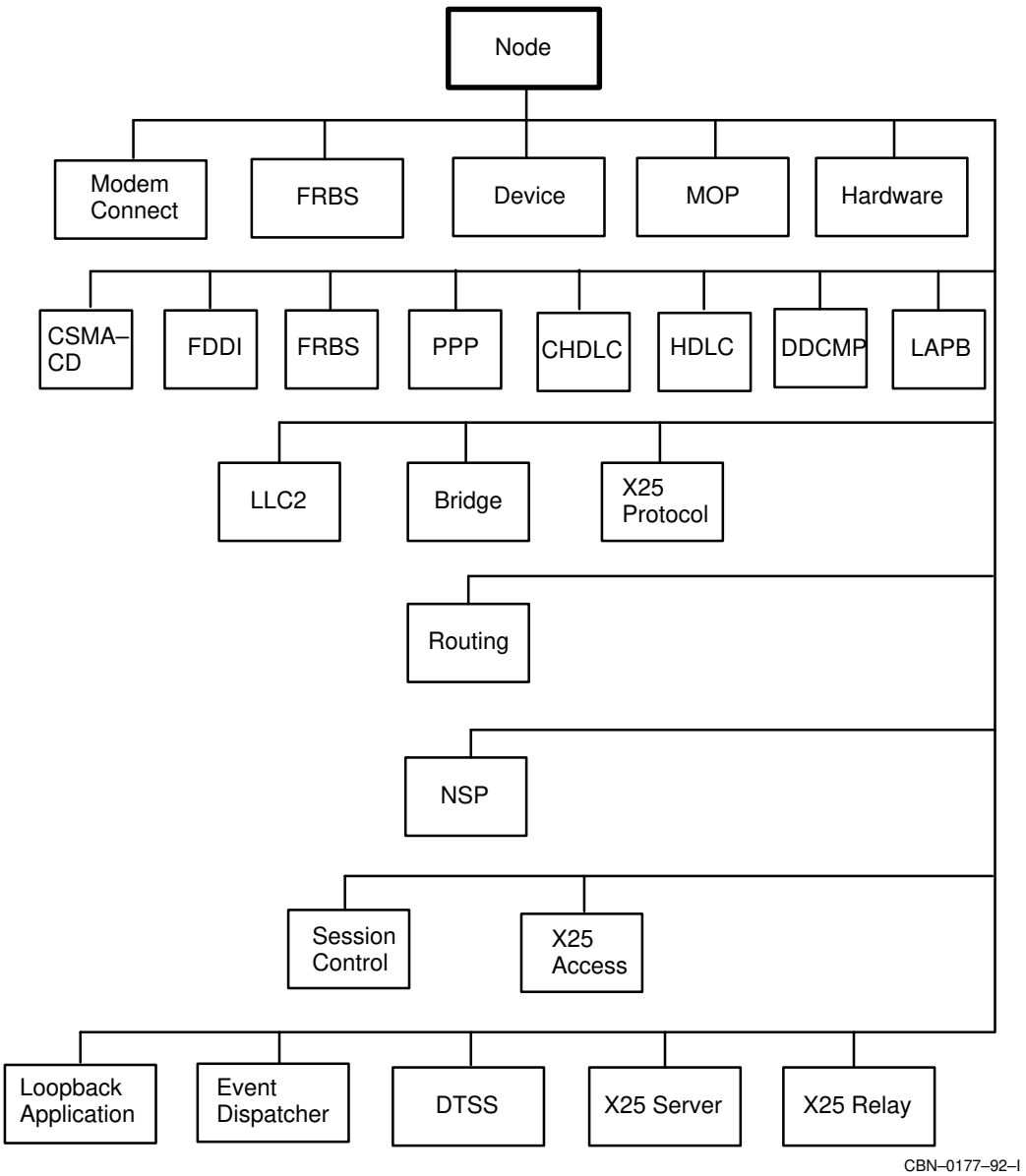
Appendix A describes all the modules and associated entities used by the DECNIS.

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

Figure 6–1, Figure 6–2, and Figure 6–3 show the entities associated with a routing circuit, and how they are related.

Figure 14–2 and Figure 14–3 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 for setting and changing the configuration parameters of a system. Issue NCL commands on a host system to manipulate directly the DECNIS modules described in Appendix A.

1.3.2 Requirement: Install DECNIS Software on Management Hosts

You should install the DECNIS software on any host system that you use to issue NCL commands to manage the DECNIS.

1.3.3 Starting NCL and Issuing Commands

Follow these steps to manage the DECNIS using NCL:

1. Log on to a suitable system in the same network as the DECNIS (usually but not necessarily a load host).
2. Start NCL on the system: the network management documentation for the operating system of the host you are using tells you how to start NCL.
3. Issue the required NCL commands: the network management documentation for your load-host operating system 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.

1.3.4 Security for NCL Commands

When you run the DECNIS configurator, you are asked for a username and password to be used for controlling access to NCL: see Section 4.2. You must use this access control information in any NCL commands you issue for the DECNIS, except for SHOW commands.

1.3.5 Example NCL Command

An example NCL command is:

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

decnis is the name of the DECNIS on which you want to create the routing circuit, SMITH is the username for the NCL application on the DECNIS, and SECRET is the password.

1.3.6 Using a Default Node

If you wish to issue several NCL commands to the same DECNIS, set default to the remote system as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

where *decnis* is the node name of the DECNIS.

1.3.7 Setting Default Security for NCL Commands

If you wish to enter several NCL commands to the same DECNIS, set default access to NCL on that system. When you have set default to the remote system (see above), issue the following command:

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

1.4 Static and Dynamic Management

1.4.1 Dynamic Management: Definition

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

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

1.4.2 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 DECNIS reverts to the permanent configuration each time it is rebooted.

1.5 NCL Script Files and CMIP Files

1.5.1 NCL Script Files

An NCL script file is an ASCII file containing NCL commands. The NCL script file resides on the load host(s) for your DECNIS, and constitutes the permanent configuration for the DECNIS.

1.5.2 CMIP Files

You must convert the NCL script file to binary CMIP (Common Management Information Protocol) format before you can downline load it to the DECNIS. When you reboot the DECNIS, the CMIP file is automatically loaded.

1.5.3 Converting an NCL Script File to a CMIP File

Use the following commands to convert the NCL script file to binary CMIP form:

For OpenVMS load hosts:

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

NCL-script is the name of the NCL script file.

For ULTRIX load hosts:

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

NCL-script is the name of the NCL script file.

1.6 Using the DECNIS Configurator to Create the NCL Script File

1.6.1 DECNIS Configurator Description

The configurator is a menu-driven program that prompts you for information about your configuration. Based on the information you supply, the configurator creates a master NCL script file and compiles it to binary CMIP format (see Section 1.5). This can then be downline loaded to the DECNIS.

1.6.2 Modifying the Permanent Configuration Using the Configurator

Follow these steps to use the configurator:

1. Specify Modify from the Main Menu of the 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. Now 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 from the NCL script

Select this if you do not want to make further changes to the permanent configuration by editing the user NCL script files (see Section 1.7). When the configurator has created the CMIP file, you will be returned to the Main Menu, from which you should select the Exit From This Procedure option.

- Return to Sections Menu

  Select this if you want to check some of the information you provided previously, or if you wish to 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. Select Create or Modify from the Main Menu, and specify the name of the DECNIS.

- Exit From the Configurator

  Select this if you want to make further changes to the permanent configuration by editing the user NCL script files (see Section 1.7). Edit the user NCL script files, and compile the master NCL script file into the CMIP file by following the instructions in Section 1.5.3.

1.7 User NCL Script Files

1.7.1 Introduction

The NCL script file produced by the configurator (see Section 1.6) automatically invokes three additional, user NCL script files. These are generated, and left empty, when the configurator is first run. You can enter NCL commands in these files to change the permanent configuration of the DECNIS.

Warning

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

1.7.2 Types of User NCL Script File

There are three user NCL script files, as follows:

- NIS_*client-name*_EXTRA_CREATE.NCL

  Add NCL CREATE commands to this file, to create new entities.

- NIS_*client-name*_EXTRA_SET.NCL

  Add NCL commands to this file that change the values of an entity's characteristic attributes, such as SET, ADD, REMOVE, and BLOCK.

- NIS_*client-name*_EXTRA_ENABLE.NCL

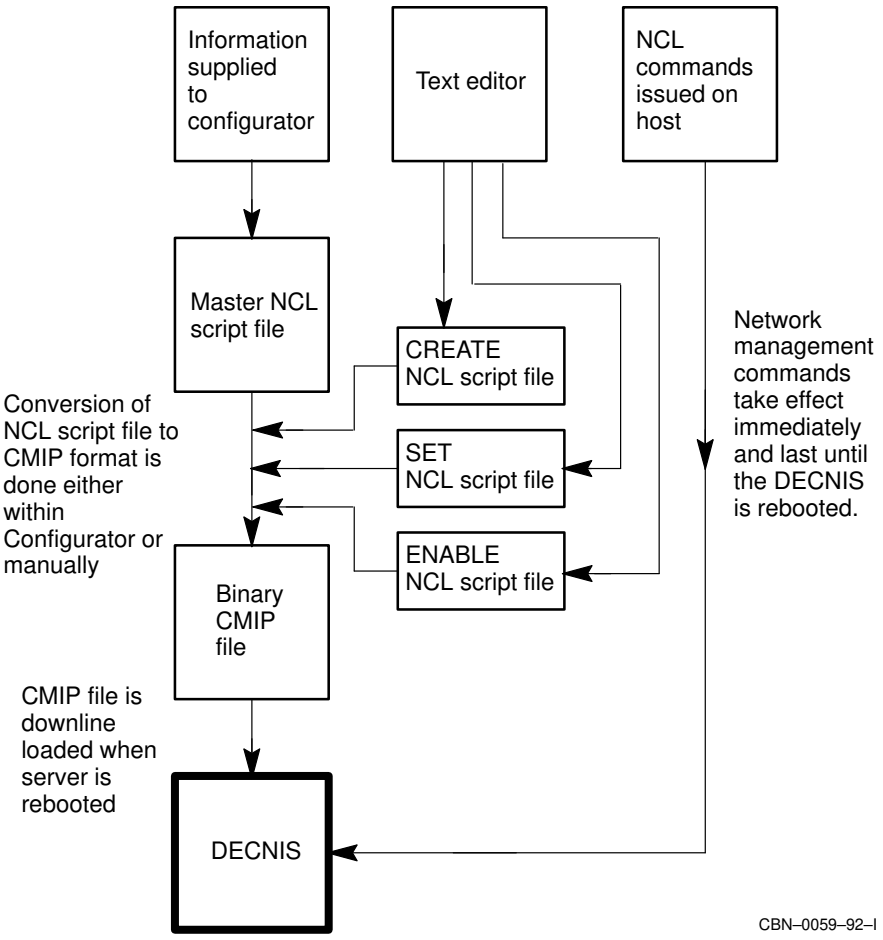
  Add NCL ENABLE commands to this file, to enable the entities created by commands in NIS_*client-name*_EXTRA_CREATE.NCL.

In the above file names, *client-name* is the name you use to identify the DECNIS when you run the load-host configurator.

1.7.3 How the DECNIS Uses the User NCL Script Files

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

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



1.7.4 Example

To use a nondefault value for the acknowledge timer on a particular HDLC data link:

1. Determine the name of the HDLC link, by examining the master NCL script file.
2. Determine the command required to change the value of the acknowledge timer (see Section 5.10); this command is:

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

3. Insert this command in the NIS_*client-name*_EXTRA_SET.NCL file.
4. Compile the master NCL script file to binary CMIP form.
5. Reboot the DECNIS.

1.8 DECdns

1.8.1 Introduction

Your network may use DECdns as a naming service. 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.

The DECNIS does not contain DECdns clerk software.

The configurator, if running on a host system that is running DECdns clerk software, can use DECdns to build the Known Towers database on the DECNIS (see Section 1.9), and can supply information about the address(es) of the DECNIS to DECdns.

See the DECdns documentation for more information about DECdns.

1.8.2 Local Naming Option (LNO)

The Local Naming Option (LNO) is an alternative to using DECdns servers in a network. Using LNO, each node in a network maintains a discrete, local namespace, containing addressing information about the other nodes in the network.

The configurator, if running on a host system that is using the LNO, can automatically build the Known Towers database on the DECNIS.

1.9 Translation of Node Names on the DECNIS

1.9.1 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 3.3 you enter the node name of the event sink).

For each node name you need to enter, 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 to translate node names into addressing information, since the DECNIS does not have DECdns clerk software.

1.9.2 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.

1.10 Entering Tower Sets in the Known Towers Database

1.10.1 Introduction

You can either use the configurator to do this automatically, or you can issue NCL commands interactively.

1.10.2 Entering Tower Sets in the Known Towers Database Using the Configurator

Using DECdns

If the host on which you are running the configurator is running DECdns clerk software, and there is a suitable DECdns server, you can choose to enter DECdns node names where necessary, for example, when specifying an event sink.

In this case, the configurator will use DECdns to find the necessary towers, and will create entries in the NCL master script file to add them to the Known Towers database.

Using the Local Naming Option (LNO)

If the host on which you are running the configurator is using LNO, you can choose to enter node names as defined in the local namespace where necessary, for example, when specifying an event sink.

In this case, the configurator will find the towers from the local namespace, and will create entries in the NCL master script file to add them to the Known Towers database.

Not Using DECdns or LNO

If you do not want to use DECdns or LNO, you can specify NSAP or Phase IV addresses where necessary, instead of node names. The configurator will construct the relevant towers, and will create entries in the master NCL script file to add them to the Known Towers database.

1.10.3 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 towers 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.

1.10.4 Tower Structure

Each tower has the following form:

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

n	is 3 if the tower refers to a DECnet/OSI node. is 2 if the tower refers to a Phase IV node.
NSAP-address	is the NSAP address of the node to which the tower refers. You can specify the NSAP address in Digital format or in OSI format (see Section 9.8).

1.10.5 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]) -
_NCL> }
```

where *decnis* is the name of the DECNIS.

1.11 Creating MOP Circuits for Loopback Testing

1.11.1 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 online Problem Solving manual on the Bookreader 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 the online Problem Solving manual on the Bookreader, you must create a MOP circuit for the relevant data link, and include LOOP REQUESTER as one of its functions.

This is done automatically when you use the configurator to create new data links.

PPP and CHDLC Data Links

You cannot set up a MOP circuit on a PPP or CHDLC data link.

1.11.2 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
   - LAPB

   circuit-name identifies the MOP circuit. For simplicity, and consistency with MOP circuits created by the configurator, 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. Enable the LOOP REQUESTER function on the MOP circuit:

```
NCL> ENABLE NODE decnis MOP CIRCUIT circuit-name FUNCTION {LOOP REQUESTER}
```

1.12 DECmcc

DECmcc™ (now integrated into the POLYCENTER™ family of products) is a comprehensive system of integrated network management software that can be used to manage the DECNIS. DECmcc allows you to monitor, manipulate, and test the modules and entities of the management model (see Appendix A).

The commands described in this manual can be entered using DECmcc instead of NCL, either by entering the DECmcc commands directly or by using the DECmcc DECwindows™ interface. In addition, other functions are available from DECmcc.

You must use Version 1.2 or later of DECmcc to manage this version of the DECNIS.

If you have DECmcc installed when you install the DECNIS software, the installation will customize the Iconic Map window to add the Load Host and DECNIS configurators to the Applications menu.

Refer to the documentation for DECmcc for details of how to use DECmcc to manage the DECNIS.

2

Configuring SNMP on the DECNIS

2.1 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.

2.2 MIBs

2.2.1 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.

2.2.2 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.

2.3 Overview

Figure 2–1 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

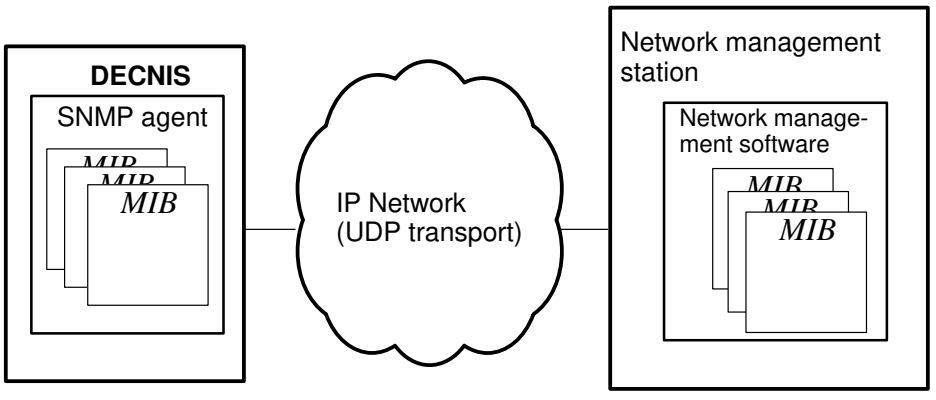
This is the SNMP-based management application running on the network management station.

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 2–1 Managing the DECNIS Using SNMP



CBN–0069–93–I

2.4 Access Control

2.4.1 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.

2.4.2 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.

2.5 MIBs Supported by the DECNIS

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

- DEC Vendor MIB elanext V2.7

  MIB groups that the DECNIS supports in this MIB are:

  - efddi

    efddiSMT
    efddiMAC
    efddiPORT
    efddiFDX

  - ebridge
  - eauth

2.5.1 Location of MIBs

The DEC Vendor MIB elanext V2.7 is supplied with the DECNIS kit: the file name and location is specific to your operating system.

In addition, you can get this MIB from the Internet at gatekeeper.dec.com through anonymous ftp. Change directory to /private/mib, look at the README file for the list of file names, and copy the appropriate file by name.

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.

2.6 Traps

2.6.1 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.

2.6.2 Traps Supported by the DECNIS

The DECNIS sends the traps specified in Table 2–1. Note that all traps include the IP address of the DECNIS.

Table 2–1 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.
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.

2.7 Configuring the DECNIS to Be Manageable Using SNMP

2.7.1 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 to set up SNMP management. In this case, you must reboot the DECNIS before the changes will take effect.

2.7.2 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
```

2.7.3 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
```

2.7.3.1 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.

2.7.4 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 those systems to which the DECNIS will send traps, together with the community name to be included in traps sent by the DECNIS.

   ```
   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 that will 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 in traps.

   You may enter up to four IP address/community name combinations, but remember that 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.

2.7.5 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 2–2 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 2–2 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.

2.8 Using SNMP to Manage the DECNIS

Section 5.23 describes the use of SNMP to manage data links on the DECNIS.

Section 7.20 describes the use of SNMP to manage the IP routing functions of the DECNIS.

Section 14.17 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.

3

Setting Up Event Logging on the DECNIS

3.1 Event Logging on the DECNIS

3.1.1 Outbound Event Streams

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

3.1.2 Example

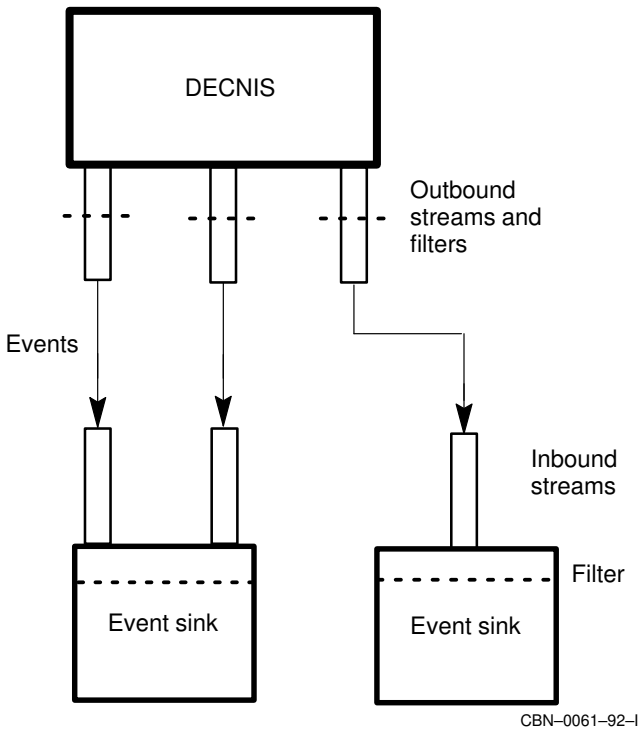
Figure 3–1 illustrates a DECNIS on which three event streams have been set up, to allow events to be passed to two event sinks.

3.2 Event Sinks

3.2.1 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 3–1 Event Streams on the DECNIS



3.2.2 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 1.9) 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.

3.2.3 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.

3.2.4 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.

3.3 Setting Up Event Logging

3.3.1 Introduction

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

3.3.2 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 1.9).

   #### 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
   ```

3.4 Disconnecting the DECNIS from the Event Sink

3.4.1 Introduction

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

3.4.2 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.

3.4.3 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.

3.4.4 Reestablishing the Connection

The DECNIS will attempt to reestablish the connection automatically when the CONNECT RETRY TIMER expires (see Section 3.6).

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

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

3.5 Disabling Event Streams

3.5.1 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.

3.5.2 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 3.4.2).

- 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 3.4.3). 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
```

3.6 Connection Timers

3.6.1 Introduction

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

3.6.2 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.

3.6.2.1 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.

3.6.3 Disconnect Timer

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

3.6.3.1 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.

3.7 Event Filtering on the DECNIS

3.7.1 Introduction

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

3.7.2 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.

3.7.3 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.

3.7.4 Catchall Filter

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

3.7.5 Operation of Event Filters

Figure 3–2 illustrates how event filters operate on the DECNIS.

3.7.6 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.

3.8 Events Blocked by Default

By default, global filters are set to block the events shown in Table 3–1 on all DECNIS outbound streams.

Figure 3–2 Operation of Event Filters in the Outbound Stream

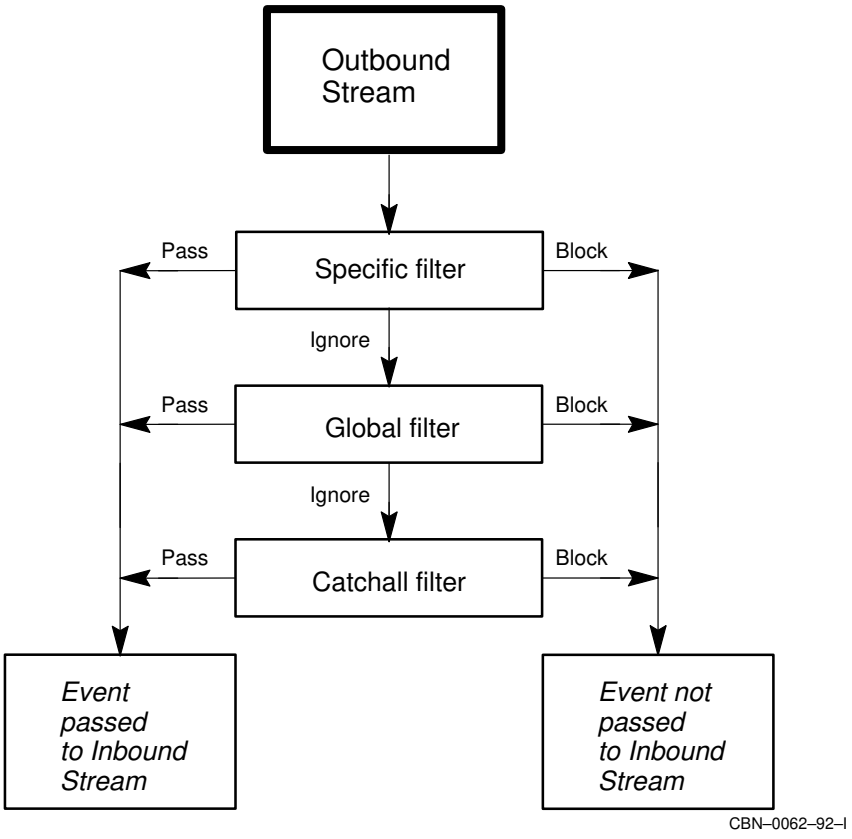


Table 3–1 DECNIS Events Blocked by Default

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

3.8.1 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 3.10.4, or you can override the blocks for specific instances of the entity, as described in Section 3.9.4.

3.9 Setting Up Specific Event Filters

3.9.1 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.

3.9.2 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.

3.9.3 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)
```

3.9.4 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.

3.9.5 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)
```

3.10 Setting Up Global Event Filters

3.10.1 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.

3.10.2 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.

3.10.3 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.

3.10.4 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.

3.10.5 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.

3.11 Setting Up a Catchall Event Filter

3.11.1 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.

3.11.2 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.

3.11.3 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.

3.12 Removing Specific and Global Event Filters

3.12.1 Introduction

This section describes how to use the IGNORE command to remove specific or global event filters.

3.12.2 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 3.7.3).

3.12.3 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 3.7.4), 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.

3.13 Testing Event Streams and Filters

3.13.1 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.

3.13.2 Command Structure

The TESTEVENT command is as follows:

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

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.

3.13.3 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
```

4

System-Level Management

4.1 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

4.2 Security for NCL

4.2.1 Need for Security

Security for the use of NCL is required to prevent unauthorized changes to the DECNIS.

4.2.2 NCL Security Mechanism

The use of NCL commands to alter the system configuration is restricted by creating a username and password on the management listener, OBJ_19. You supply this username and password when you use the configurator to configure the DECNIS. However, you cannot restrict the use of the NCL SHOW command.

4.2.3 Changing the Username and Password for NCL

To change a username and password, enter the following command:

```
NCL> SET NODE decnis SESSION CONTROL APPLICATION obj_19 -
_NCL> USER NAME username, PASSWORD password
```

where:

decnis	is the name of the DECNIS.
username and *password*	are the username and password that users must quote when they use NCL on the DECNIS.

Result

To use NCL on the DECNIS, users will need to quote the username and password in the usual way for their operating system.

4.2.4 Example

To create a CSMA/CD circuit called SALT on a node called ORG:.SOUTH.SALES, which has a username SMITH and password SECRET, enter the following commands:

- For an OpenVMS host:

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

- For an ULTRIX or PC host:

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

4.3 Security for the Common Trace Facility

4.3.1 Need for Security

Security for the Common Trace Facility (CTF) is required to prevent unauthorized users from monitoring the data passing through the DECNIS.

4.3.2 CTF Security Mechanism

Access to the Common Trace Facility (CTF) is restricted by creating a username and password on the CTF listener, OBJ_54. You supply this username and password when you use the configurator to configure the DECNIS.

4.3.3 Changing the Username and Password for CTF

To change the username and password on the CTF listener, enter the following command:

```
NCL> SET NODE decnis SESSION CONTROL APPLICATION obj_54 -
_NCL> USER NAME username, PASSWORD password
```

where:

decnis	is the name of the DECNIS.
username and *password*	are the username 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 username and password in the usual way for their operating system.

4.3.4 Example

To start CTF on a node called ORG:.SOUTH.SALES, which has a username SMITH and password SECRET, specifying a tracepoint "HDLC LINK THIS-LINK", enter:

```
$ TRACE START org:.south.sales"smith secret"::"HDLC LINK this-link"
```

4.4 Preventing Unauthorized Loading

4.4.1 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.

4.4.2 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.

4.4.3 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

You can clear the nonvolatile memory of the DECNIS by powering up the hardware with the Dump button held in: see the *Installation and Service Manual* for your hardware.

4.5 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).

4.6 Security for SNMP

4.6.1 Security for SNMP Requests

You can use the configurator or NCL commands to set up community names to validate SNMP Get and Set commands sent to the DECNIS. The DECNIS will not respond to SNMP requests that contain an incorrect community name.

Section 2.7 describes how to set up community names.

4.6.2 Security for Traps

You can use the configurator, NCL commands, or SNMP requests to set up a community name that will be sent in traps from the DECNIS.

The management stations to which the DECNIS traps are sent may then use this community name to reject or accept the trap.

Section 2.7 describes how to set up a community name in traps.

4.7 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
```

4.8 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. Digital recommends that you use the DECdns node name, if used.
sysLocation	Specifies the physical location of the DECNIS.

5
Connections to the Network

5.1 Introduction

This chapter describes how to create and delete data link connections between the DECNIS and the network.

5.1.1 NCL Commands

All tasks performed using NCL commands assume that you have already logged on to a suitable host system on the network and started NCL, and have set default to the DECNIS as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

5.2 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.

5.3 Port Naming Scheme

5.3.1 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 a number that identifies the slot on the DECNIS backplane into which the Network Interface Card is inserted. Slot 1 is normally occupied by the management processor and slot 2 is normally occupied by the pool memory board, leaving the remaining slots for Network Interface Cards.

port is a number identifying the port on the Network Interface Card.

5.3.2 Examples

If a LANcontroller 601 card (which has just one port) was positioned in slot 6 of the backplane, the communication port would be identified as L601-6-0.

If the card was then moved to slot 5, the communication port would be identified as L601-5-0.

If a WANcontroller 622 card (which has two ports) was positioned in slot 4 of the backplane, the two communication ports it provides would be identified as W622-4-0 and W622-4-1.

If an FDDIcontroller 621 card (which occupies two slots and has one port) was positioned in slots 5 and 6 of a DECNIS 600 backplane, the port would be identified as F621-6-0.

If the same card was positioned in slots 3 and 4 of a DECNIS 500, the port would be identified as F621-3-0.

5.4 Adding a Network Interface Card

5.4.1 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 5.5 explains live insertion in more detail.

5.4.2 Configuring the 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
```

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 concerned (see Section 5.3).

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

X is the slot in which the Network Interface Card is positioned.

F621 Network Interface Cards: Note that F621 cards occupy two slots. For the DECNIS 600 hardware platform, use the higher-numbered slot. For the DECNIS 500 hardware platform, use the lower-numbered slot.

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.

5.4.3 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. This is avoided if a DEVICE UNIT entity is created and enabled for the first line.

The configurator always creates and enables device units for both lines, avoiding this problem.

5.5 Live Insertion

5.5.1 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.

5.5.2 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.

5.5.3 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.

5.5.4 How a Card's Microcode Is Loaded

The microcode is loaded in either of the following circumstances:

- The card and its slot are specified on the Network Interface Cards screen of the DECNIS configurator, and the card is physically present in the hardware slot when the DECNIS is booted.
- The DEVICE UNIT entity representing the card is enabled on a running system, and the card is physically present in the hardware slot.

5.5.5 Restriction: F621 Network Interface Cards

You cannot insert an F621 Network Interface Card into a previously unused slot. If you attempt to do this, the card will fail to load its microcode.

5.5.6 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.

5.6 Creating a CSMA/CD Connection

5.6.1 Introduction

This section describes how to configure a CSMA/CD data link on a suitable hardware port. You can only set up CSMA/CD connections using a LAN Network Interface Card (for example, LANcontroller 601).

5.6.2 Procedure

Follow these steps to create a new CSMA/CD connection:

1. Ensure that the CSMA-CD entity exists by entering the following command:

   ```
   NCL> CREATE CSMA-CD
   ```

   Result: A message will tell you if the CSMA-CD entity 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 5.3).

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 1.11.

5.7 Creating an FDDI Connection

5.7.1 Introduction

This section describes how to configure an FDDI data link on a suitable hardware port. You can only set up FDDI connections using an FDDI Network Interface Card (FDDIcontroller 621).

5.7.2 Procedure

Follow these steps to create a new FDDI connection:

1. Ensure that the FDDI entity exists by entering the following command:

   ```
   NCL> CREATE FDDI
   ```

   Result: A message will tell you if the FDDI entity 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 5.3).

   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 1.11.

5.7.3 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 the DECNIS configurator, the ring purger is 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
```

5.8 Creating a VCP Connection

5.8.1 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.

5.8.2 Network Interface Card for VCP

You can only run VCP connections on a DEC WANcontroller 622 Network Interface Card.

5.8.3 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 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-x-y
   ```

   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 (see Section 5.3).
y	identifies the port on the Network Interface Card (0 or 1).

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 entity exists by entering the following command:

   ```
   NCL> CREATE CSMA-CD
   ```

Result: A message will tell you if the CSMA-CD entity 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 1.11.

5.8.4 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 6.3.

5.8.5 Bridging Over VCP Data Links

Section 14.5 describes how to set up a bridge port on a VCP data link.

5.9 Creating an HDLC Connection

5.9.1 Introduction

This section describes how to configure an HDLC data link on a suitable synchronous hardware port. You can only set up HDLC connections using a WAN Network Interface Card (for example, WANcontroller 622).

5.9.2 Procedure

Follow these steps to create a new HDLC connection:

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, 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 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 5.3).

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 entity 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 data link.

10. Set up a MOP circuit on the data link, to allow subsequent loopback testing: see Section 1.11.

5.10 Changing the Acknowledge and Holdback Timers on an HDLC Connection

5.10.1 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.

5.10.2 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.

5.10.3 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.

5.10.4 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.

5.10.5 Setting the Holdback Timer

Set the HOLDBACK TIMER characteristeric 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.

5.11 Changing Preferred Window Size on an HDLC Connection

5.11.1 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.

5.11.2 Guideline for Setting Preferred Window Size

Increase the value from the default (2) if there is a significant delay on the link.

5.11.3 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.

5.12 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.

5.13 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.

5.14 Creating a DDCMP Connection

5.14.1 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. You can only set up DDCMP connections using the DEC WANcontroller 614 or DEC WANcontroller 618 Network Interface Cards.

5.14.2 Procedure

Follow these steps to create a new DDCMP connection:

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, specify the hardware port on which the connection will exist, and specify 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 5.3).
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 entity 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 data link.

10. Set up a MOP circuit on the data link, to allow subsequent loopback testing: see Section 1.11.

5.15 Changing the Retransmit and Holdback Timers on a DDCMP Connection

5.15.1 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.

5.15.2 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.

5.15.3 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.

5.15.4 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.

5.15.5 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.

5.16 Changing the Transmit Window on a DDCMP Connection

5.16.1 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.

5.16.2 Guideline for Setting Transmit Window Size

The value need only be changed from the default supplied by the configurator (255) if the remote station has limited transient receive buffering. It should be set to a large enough value so that the window does not close in the time taken for an acknowledgment to arrive.

5.16.3 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.

5.17 Creating a PPP Connection

5.17.1 Introduction

This section describes how to configure a PPP data link on a suitable synchronous hardware port. You can only set up PPP connections using a WAN Network Interface Card (for example, WANcontroller 622).

If you want to set up a PPP connection over a frame relay network, follow the procedures in Section 5.19.

5.17.2 Procedure

Follow these steps to create a new PPP connection:

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, specifying the hardware port on which the connection will exist, and specifying 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 5.3).

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 entity 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
   ```

5.17.3 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.

5.18 Creating a CHDLC Connection

5.18.1 Introduction

This section describes how to configure a CHDLC data link on a suitable synchronous hardware port. You can only set up CHDLC connections using a WAN Network Interface Card (for example, WANcontroller 622).

If you want to set up a CHDLC connection over a frame relay network, follow the procedures in Section 5.19.

5.18.2 Procedure

Follow these steps to create a new CHDLC connection:

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, specifying the hardware port on which the connection will exist, and specifying 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 5.3).

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 entity exists:

```
NCL> CREATE CHDLC
```

6. Create a CHDLC data link for the circuit:

```
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.

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
```

5.19 Creating a PPP or CHDLC Connection Using Frame Relay

5.19.1 Introduction

This section describes how to configure a PPP or CHDLC data link over a frame relay network. You can only set up PPP and CHDLC connections over frame relay using a DEC WANcontroller 622 Network Interface card. This card provides high-speed lines at T1/E1 (56 kbits/s to 2.048 Mbits/s).

If you want to create a PPP link to a DEC WANrouter/Frame Relay 100/500, refer to Section 5.20.

5.19.2 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 connection is associated with 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.

5.19.3 Defining a PVC for the 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 5.19.6 describes how to assign a DLCI to a connection.

5.19.4 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 either define the protocol during the configuration procedure (refer to the *DEC Network Integration Server Installation and Configuration* manual), or you can define it when you create a frame relay channel (see Section 5.19.6).

5.19.5 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.

5.19.6 Procedure

To create a PPP or CHDLC link over frame relay, complete the following:

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, 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 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 5.3).

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 entity 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* created earlier.

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 entity exists:

```
NCL> CREATE CHDLC
```

13. If you are setting up a PPP connection, ensure that the PPP entity 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
```

5.20 Creating a PPP Connection to a DEC WANrouter/Frame Relay 100/500

5.20.1 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 5.20.3 describes how to set the Required Control Protocols attribute for the PPP link so that the DECNIS and DEC WANrouter/Frame Relay can interoperate.
- Only IP and OSI network layer packets can be sent over a PPP link to a DEC WANrouter/Frame Relay.

5.20.2 PPP Network Control Protocols

PPP consists of a number of different network control protocols. The purpose of these protocols is to negotiate the exchange of network layer packets sent over a PPP link.

There is a network control protocol defined for each type of network layer packet sent over the link. For example, an IP control protocol for IP packets, OSI Network Layer control protocol for OSI packets, and a DECnet Phase IV control protocol for DECnet Phase IV packets.

A network control protocol is automatically started when a routing circuit accesses the PPP link to transfer a type of network layer packet.

For example, if a routing circuit accesses the link to send IP packets to another system, the IP network control protocol is started. The network control protocol negotiates the exchange of IP packets with the other system, and, when various options have been proposed or rejected, IP network layer packets are sent over the link.

5.20.3 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 5.19.6 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
```

5.21 Deleting a Connection

5.21.1 Reason for Deleting

Delete a connection if you want to permanently prevent the DECNIS from communicating with the network through that connection.

5.21.2 Disabling Connections

Use the DISABLE command if you want to temporarily prevent the DECNIS from communicating with the network through a specific connection.

5.21.3 Before Deleting a Connection

Before you delete a connection, you should remove any routing, bridging, or X.25 functions that use the connection.

5.21.4 Deleting 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 5.22 if you do not know the name of the CSMA-CD STATION):

```
NCL> DISABLE CSMA-CD STATION station-name
```

5.21.5 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
```

5.21.6 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 5.22 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
   ```

5.21.7 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 5.22 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
```

5.21.8 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 5.22 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
```

5.21.9 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 5.22 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
```

5.22 Identifying Management Entities from the Hardware Port

5.22.1 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
- HDLC LINK
- HDLC LINK LOGICAL STATION
- DDCMP LINK
- DDCMP LOGICAL STATION
- PPP LINK
- FRBS CHANNEL
- FRBS CHANNEL CONNECTION

5.22.2 CSMA-CD STATION Name

Issue the following command to display the name of the CSMA-CD STATION entity:

```
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.

5.22.3 FDDI STATION Name

Issue the following command to display the name of the FDDI STATION entity:

```
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.

5.22.4 MODEM CONNECT LINE Name

Issue the following command to display the name of the MODEM CONNECT LINE entity:

```
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.

5.22.5 HDLC LINK Name

Issue the following command to display the name of the HDLC LINK entity:

```
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 5.22.4.

5.22.6 HDLC LINK LOGICAL STATION Name

Issue the following command to display the name of the HDLC LINK LOGICAL STATION entity:

```
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 5.22.5.

5.22.7 DDCMP LINK Name

Issue the following command to display the name of the DDCMP LINK entity:

```
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 5.22.4.

5.22.8 DDCMP LOGICAL STATION Name

Issue the following command to display the name of the DDCMP LINK LOGICAL STATION entity:

```
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 5.22.5.

5.22.9 FRBS CHANNEL Name

Issue the following command to display the name of the FRBS CHANNEL entity:

```
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 5.22.4.

5.22.10 FRBS CHANNEL CONNECTION Name

Issue the following command to display the name of the FRBS CHANNEL CONNECTION entity:

```
NCL> SHOW FRBS CHANNEL channel-name CONNECTION * -
_NCL> ALL IDENTIFIERS
```

where *channel-name* is the name of the associated channel, as discovered in Section 5.22.9.

5.22.11 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 5.22.4.

- 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 5.22.9, and *connection-name* is the name of the connection, as discovered in Section 5.22.10.

5.23 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 configurator to set up the data links, the name will be the same as the hardware port (see Section 5.3).

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
CHDLC	ppp
CSMA/CD	ethernet-csmacd
DDCMP	other
FDDI	fddi
HDLC	propPointToPointSerial
LAPB	lapb
PPP	ppp
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 6 describes how to set up routing circuits, and discusses routing security.
- Chapter 7 describes how to use the DECNIS for IP routing.
- Chapter 8 describes how the DECNIS implements IP route propagation and filtering.
- Chapter 9 describes how to use the DECNIS for DECnet/OSI routing.
- Chapter 10 describes how to use the DECNIS for routing AppleTalk and NetWare IPX protocols.

6

Managing Routing: General

6.1 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, and X.25 routing circuits
- Routing level
- Routing security

6.1.1 NCL Commands

All tasks performed using NCL commands assume that you have already logged on to a suitable host system on the network and started NCL, and have set default to the DECNIS as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

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.

6.2 Entities Associated with a Routing Circuit

Figure 6–1, Figure 6–2, and Figure 6–3 show the entities required by each type of routing circuit.

6.3 Adding a CSMA/CD Routing Circuit

6.3.1 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 5.6 and Section 5.8.

6.3.2 Procedure

Create a new CSMA/CD routing circuit as follows:

1. Create a routing circuit, specifying that it is 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:

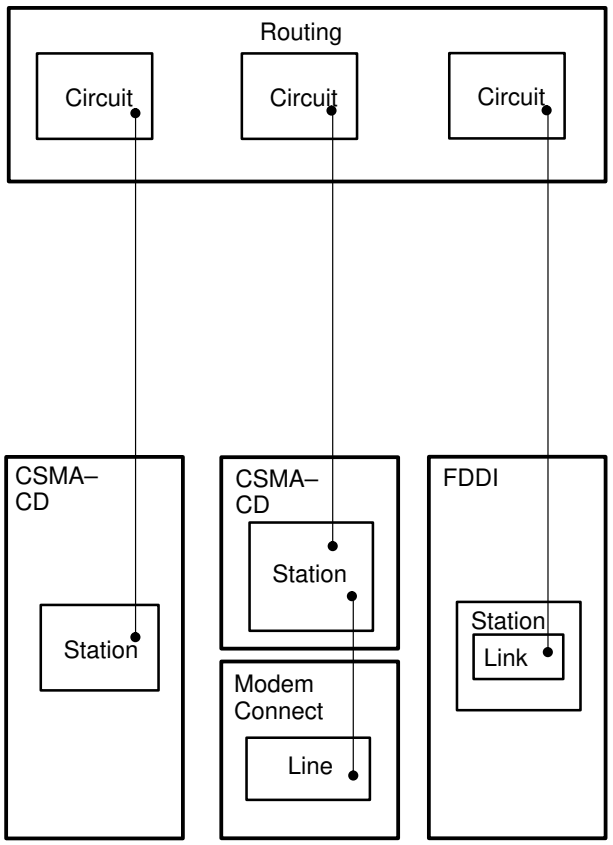
   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

6.4 Adding an FDDI Routing Circuit

6.4.1 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 5.7.2.

Figure 6–1 Entities Required by LAN and VCP Routing Circuits

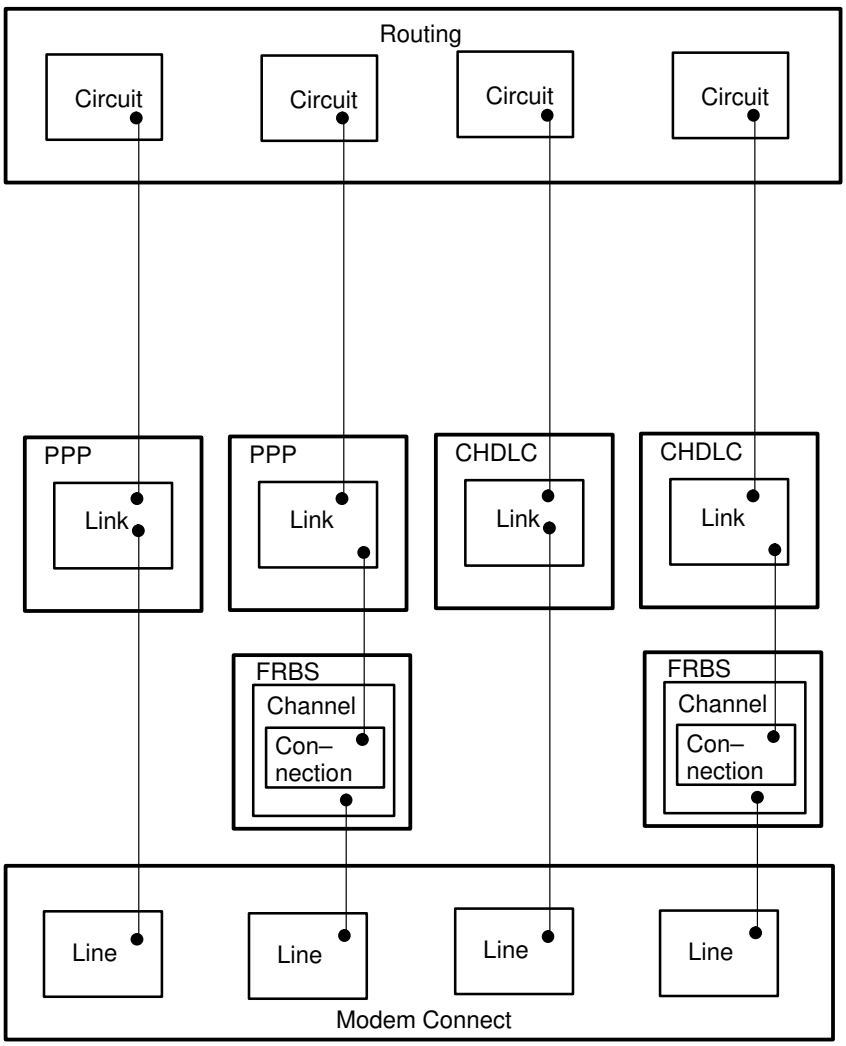


An association between two entities (indicated by a line connecting them) is made by specifying the name of the lower entity as a characteristic value of the upper entity.

For example: a routing circuit of type FDDI specifies as its DATA LINK entity the name of the FDDI link.

CBN–0070–93–I

Figure 6–2 Entities Required by PPP and CHDLC Routing Circuits

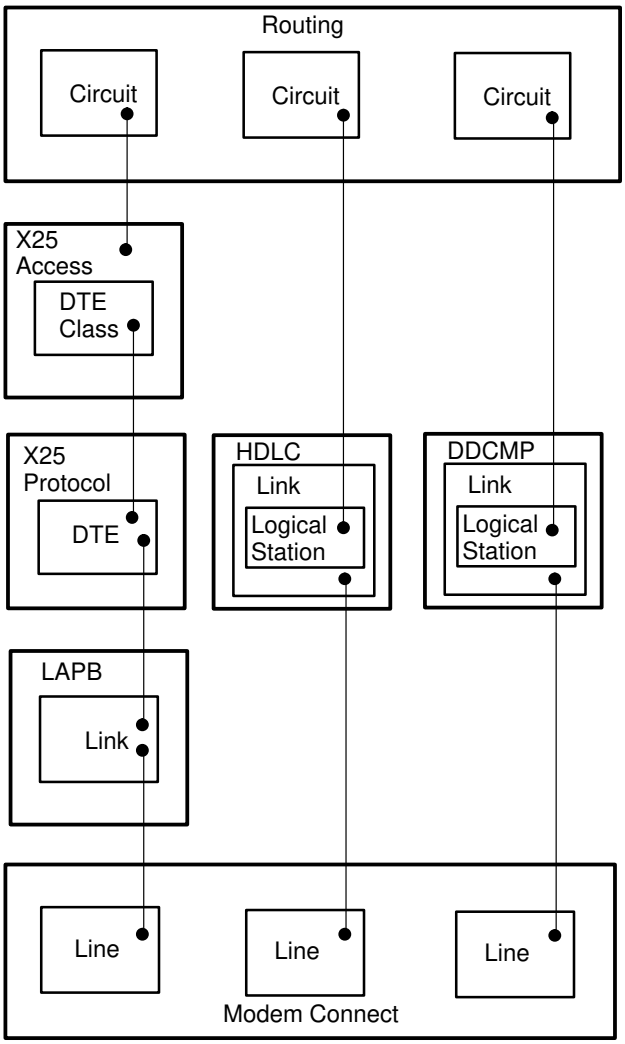


An association between two entities (indicated by a line connecting them) is made by specifying the name of the lower entity as a characteristic value of the upper entity.

For example: a routing circuit of type PPP specifies as its DATA LINK entity the name of the PPP Link.

CBN–0071–93–I

Figure 6–3 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–0072–93–I

6.4.2 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
   ```

6.5 Adding an HDLC or DDCMP Routing Circuit

6.5.1 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 5.9 and Section 5.14.

6.5.2 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
   ```

6.6 PPP Routing Circuits

6.6.1 Introduction

PPP 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 when the DECNIS is configured (see *DEC Network Integration Server Installation and Configuration*) or when a circuit is added (see Section 6.7).

6.6.2 Defining Protocols for PPP 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 all IP data and IP routing control packets. For example, EGP and RIP.
- **ISO8473** for all DECnet/OSI data and DECnet/OSI routing control packets. For example, integrated IS-IS (ISO 10589/RFC 1195) and ES-IS (ISO 9542).
- **DECNET PHASE IV** for all DECnet Phase IV data and DECnet Phase IV routing control packets.

For example, if the remote system uses IP routing, you must define IP as the network protocol routed over the circuit between the DECNIS and remote system.

6.6.3 Default Protocols

If you do not define a protocol for the circuit, the system sets up IP and ISO8473 on the circuit by default.

6.6.4 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.

6.6.5 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.

6.7 Adding a PPP Routing Circuit

6.7.1 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 5.17.

6.7.2 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. 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

   For example:

   ```
   NCL> SET ROUTING CIRCUIT W622-4-1 NETWORK PROTOCOLS {IP, DECNET PHASE IV}
   ```

If you do not use this command to specify a protocol, IP and ISO8473 are set up on the circuit automatically.

4. Enable the circuit:

```
NCL> ENABLE ROUTING CIRCUIT circuit-name
```

6.8 CHDLC Routing Circuits

6.8.1 Introduction

CHDLC 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 when the DECNIS is configured (see *DEC Network Integration Server Installation and Configuration*) or when a circuit is added (see Section 6.9).

CHDLC routing circuits are PPP circuits running over the CHDLC data link. CHDLC routing circuits have the same conditions attached to them as PPP routing circuits. See Section 6.6.2 to Section 6.6.5 for background information on PPP Routing Circuits.

6.9 Adding a CHDLC Routing Circuit

6.9.1 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 5.18.

6.9.2 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. 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

For example:

```
NCL> SET ROUTING CIRCUIT W622-4-1 NETWORK PROTOCOLS {IP, DECNET PHASE IV}
```

If you do not use this command to specify a protocol, IP and ISO8473 are set up on the circuit automatically.

4. Enable the circuit:

```
NCL> ENABLE ROUTING CIRCUIT circuit-name
```

6.10 Changing the Phase IV Routing Vector Timer on PPP and CHDLC Circuits

CHDLC routing circuits are PPP circuits using the CHDLC data link, so the same conditions for timer settings apply.

6.10.1 Introduction

The Phase IV Routing Vector Timer sets the time between each routing vector packet that is transmitted over a PPP circuit. It is only available on PPP circuits.

The Phase IV Routing Vector Timer is useful if, for example, you want to increase the frequency of routing vector updates on PPP circuits 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.

6.10.2 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 65535. For example:

```
NCL> SET ROUTING CIRCUIT W622-4-1 PHASE IV ROUTING VECTOR TIMER 15
```

6.11 X.25 Routing Circuits

6.11.1 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).

6.11.2 Types of X.25 Routing Circuit

There are two main types of X.25 routing circuit, as described in Table 6–1.

Table 6–1 Types of X.25 Routing Circuit

Type	Description
Dynamically Assigned	An X.25 virtual circuit is only created when there is traffic to send or receive. Only Level 2 routers can have this type of X.25 circuit. Adaptive routing information is not exchanged across this type of X.25 circuit.
Static	Static X.25 circuits form permanent connections between two DECnet routers. Routing information is exchanged across this type of X.25 circuit, in the same way as across an HDLC routing circuit. There are three types of static X.25 routing circuit:

Name	Description
Outgoing	The DECNIS initiates the X.25 call used to establish the connection.
Incoming	The DECNIS accepts the X.25 call from the remote DECnet router.
Permanent	The DECNIS is connected to the remote DECnet router by means of a PVC (Permanent Virtual Circuit).

6.12 Templates for X.25 Routing Circuits

6.12.1 Template Function

An X.25 routing circuit uses a template to set the X.25 call characteristics for outgoing calls.

Section 6.13 explains how to create templates, and specify the call characteristics that they will set.

6.12.2 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)

6.12.3 Template Characteristics

Table 6–2 shows the template characteristics required for each type of X.25 routing circuit.

Table 6–2 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.

(continued on next page)

Table 6–2 (Cont.) Template Characteristics for X.25 Routing Circuits

Circuit Type	Template Characteristic	(R)equired or (O)ptional	Value
	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.

6.13 Creating and Deleting X.25 Templates

6.13.1 Introduction

This section describes how to create a template and specify the characteristics of the outgoing calls that it will set.

6.13.2 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.

6.13.3 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
```

6.14 Filters for X.25 Routing Circuits

6.14.1 Introduction

An X.25 routing circuit uses a filter to select incoming calls that are intended for that circuit.

Section 11.5 describes the operation of filters, and how to create and manage them.

6.14.2 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

6.14.3 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 13.2 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 be set to 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 be set to 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
```

6.15 Adding an X.25 Dynamically Assigned Routing Circuit

6.15.1 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.

6.15.2 Routing Level

The DECNIS must be configured as a Level 2 router if it has any X.25 Dynamically Assigned circuits.

6.15.3 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 11.5.2.

   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 filter for each type of routing. 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 6.13. 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 of routing circuit required:

   ```
   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 6.13.2.

7. Set up the manual routing required for this circuit: see Section 9.14 and Section 7.12. (All routing over X25 DA circuits is manual.)

8. Enable the circuit:

   ```
   NCL>  ENABLE ROUTING CIRCUIT circuit-name
   ```

6.16 Adding an X.25 Static Outgoing Routing Circuit

6.16.1 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.

6.16.2 Procedure

Create an X.25 Static Outgoing routing circuit as follows:

1. Set up a template for outgoing calls (see Section 6.13). 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 of routing circuit required:

   ```
   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
   ```

6.17 Adding an X.25 Static Incoming Routing Circuit

6.17.1 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.

6.17.2 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 6.13.
3. Create a routing circuit, specifying the type of routing circuit required:

   ```
   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
   ```

6.18 Adding an X.25 Permanent Routing Circuit

6.18.1 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.

6.18.2 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 11.6.2.
2. Create a routing circuit, specifying the type of routing circuit required:

   ```
   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
   ```

6.19 Changing Routing Circuit Characteristics

6.19.1 Introduction

This section describes some circuit characteristics that you may wish to alter from their default values.

Each of the following subsections specifies:

- Type of routing circuit to which the characteristic applies
- Description of the characteristic
- How to change it

6.19.2 Changing the Maximum Data Link Message Size

Routing circuit type: HDLC, DDCMP, PPP, CHDLC, FDDI X.25

6.19.2.1 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.

6.19.2.2 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
   ```

6.19.2.3 FDDI Routing Circuits

For FDDI routing circuits, the DECNIS configurator sets the routing circuit characteristic MANUAL DATA LINK SDU SIZE to 1492, to ensure operability with DECnet Phase IV systems.

You can increase this value up to 4481, but other systems in the network may not support such large packet sizes.

6.19.3 Changing the Originating Queue Limit

Routing circuit type: CSMA/CD, HDLC, DDCMP, X.25, FDDI

6.19.3.1 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 alter 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).

6.19.3.2 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.

6.19.4 Changing the Recall Timer

Routing circuit type: PPP, X25 DA, X25 STATIC OUTGOING

6.19.4.1 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 6.19.5) before succeeding in making the call.

6.19.4.2 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.

6.19.5 Changing the Maximum Call Attempts

Routing circuit type: X25 STATIC OUTGOING

6.19.5.1 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.

6.19.5.2 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.

6.19.6 Changing the Maximum SVC Adjacencies

Routing circuit type: X25 DA

6.19.6.1 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.

6.19.6.2 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.

6.19.7 Changing the Idle Timer

Routing circuit type: X25 DA

6.19.7.1 Description

The idle timer indicates the time, in seconds, for which data traffic on the circuit must be absent before the call is cleared.

6.19.7.2 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.

6.19.8 Changing the Initial Minimum Timer

Routing circuit type: X25 DA

6.19.8.1 Description

This indicates the minimum time, in seconds, that a call on a circuit will remain connected, irrespective of traffic.

6.19.8.2 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.

6.20 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.

6.21 Deleting a Routing Circuit

6.21.1 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).

6.21.2 Deleting Non-X.25 Routing Circuits

Delete a CSMA/CD, FDDI, PPP, HDLC, or DDCMP 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
```

6.21.3 Deleting X.25 Routing Circuits

Delete an X.25 routing circuit as follows:

1. Follow the steps in Section 6.21.2.
2. Delete the templates associated with this routing circuit, unless they are being used by another routing circuit: see Section 6.13.3.
3. Delete any filters associated with this routing circuit, unless they will be used by another routing circuit: see Section 11.5.3.
4. Delete the DTE, if it will not be used for any other functions: see Section 11.2.9.

6.22 Changing the Routing Level of the DECNIS

6.22.1 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 to destination 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.

Consequences of Changing Routing Level

Changing the routing level of the DECNIS from Level 2 to Level 1 will cause the configurator to delete any X.25 dynamically assigned circuits, any IP or OSI reachable addresses, and any 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.

6.22.2 Cannot Change Routing Level of a Running System

To change the routing level of your DECNIS, you must run the configurator and reboot the system. You cannot change the routing level of a running system.

6.22.3 Procedure

Change the routing level of the DECNIS by supplying the following information to the configurator:

	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.

6.23 Preventing Routing Circuits from Routing Level 1 Traffic

6.23.1 Introduction

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.

6.23.2 Procedure

Prevent a routing circuit from routing Level 1 traffic as follows:

1. Disable the routing circuit:

   ```
   NCL> DISABLE ROUTING CIRCUIT circuit-name
   ```

2. Set the MANUAL L2ONLY MODE characteristic of the routing circuit to TRUE:

   ```
   NCL> SET ROUTING CIRCUIT circuit-name MANUAL L2ONLY MODE true
   ```

3. Enable the routing circuit:

   ```
   NCL> ENABLE ROUTING CIRCUIT circuit-name
   ```

6.23.3 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 9.11 and Section 9.13, respectively).

6.24 Routing Security

6.24.1 Introduction

This section describes how to secure the routing function of your DECNIS, to prevent an unauthorized adjacent system from setting up a circuit to connect with it. It applies only to HDLC, DDCMP, and and X.25 STATIC routing circuits. Section 13.2 describes how to use X.25 security for other types of X.25 routing circuit.

6.24.2 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.

6.24.3 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.

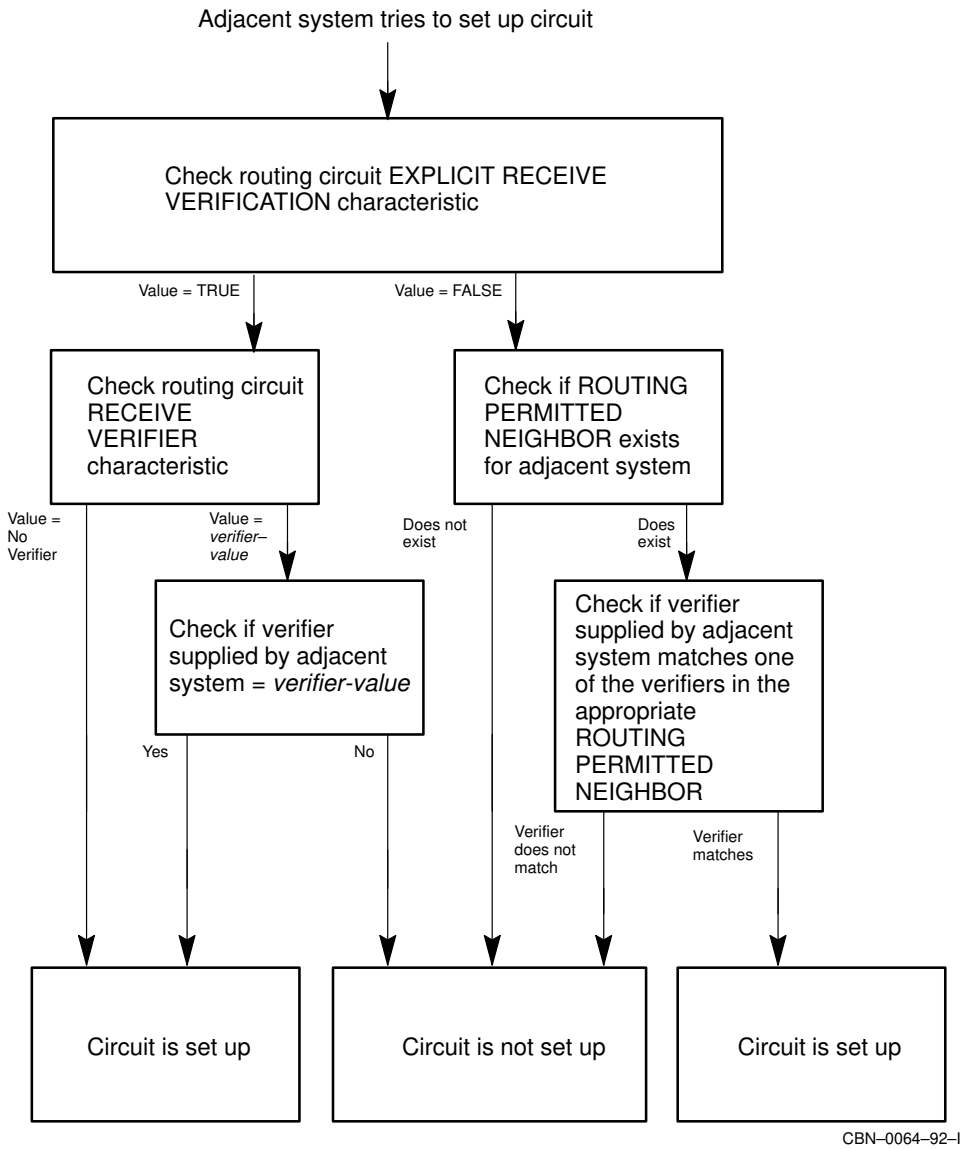
  This method is useful if there are several systems that can reach your DECNIS on the same circuit; for example, an X.25 or a dialup circuit.

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.

6.24.4 How Routing Security Works

When an adjacent system tries to set up a circuit to the DECNIS, the checks shown in Figure 6–4 are made.

Figure 6–4 Security Checks Made by Routing



6.25 Setting Up Circuit-Level Routing Security

6.25.1 Introduction

This section describes how to set up a verifier (password) for each 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.

6.25.2 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. Now set the routing 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, you should disable and enable the circuit after issuing them.

6.26 Setting Up System-Level Routing Security

6.26.1 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.

6.26.2 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 by issuing 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 9.2 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.

6.27 Configuring the DECNIS to Send Verifiers

6.27.1 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.

6.27.2 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.

6.28 Optimizing the Routing Database Sizes on the DECNIS

6.28.1 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.

6.28.2 How Routing Database Size Is Implemented on the DECNIS

The ROUTING entity has a number of status attributes and characteristic attributes, as shown in Table 6–3.

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 characteristics, refer to Section 6.28.3.

Table 6–3 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 6–3 (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 6.28.3.1.	
L2 CONNECTIVITY	A value describing the Level 2 network topology. See Section 6.28.3.1.	
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 6–3 (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 6.28.3.1.	
IP L2 CONNECTIVITY	A value describing the Level 2 IP network topology. See Section 6.28.3.1.	

6.28.3 OSPF Routing Characteristics

The following table shows the characteristics 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 access the DECNIS Configurator to change database sizes. Refer to the DECNIS online Problem Solving manual on the Bookreader to find out how to do this.

Table 6–4 OSPF General 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.

(continued on next page)

Table 6–4 (Cont.) OSPF General Control Protocol Characteristic Attributes

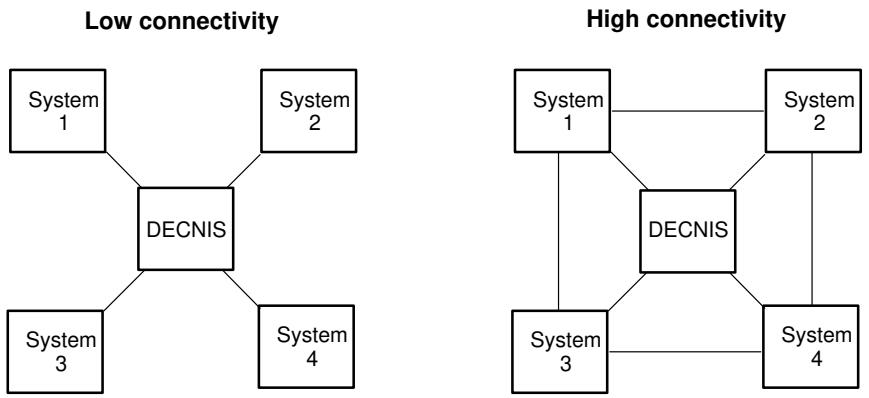
Characteristic	Explanation
OSPF MAXIMUM AREA INTERFACES	Maximum number of OSPF Interfaces to a single area on any OSPF router in a connected area.
OSPF AVERAGE AREA NETWORKS	Average number of OSPF transit and stub networks in each area that the router is connected to directly.
OSPF MAXIMUM NETWORK ROUTERS	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.

6.28.3.1 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 6–5 illustrates how a simple network can be connected to give different connectivity factors.

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.

Figure 6–5 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.

Characteristic Attributes for Connectivity

Note that the connectivity of a network as described by the connectivity characteristics in Table 6–3 is one-tenth the value of the attribute.

Example: If the Level 1 IP network connectivity of your network is approximately 4, you should set the value of the IP L1 CONNECTIVITY to 40.

6.28.4 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 6–5.

6.28.5 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 6–5.

Digital recommends that you do not alter the routing database sizes except in the following circumstances:

- You know that the default values of one or more databases are insufficient. 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.

6.28.5.1 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.

6.28.6 How to Change the Routing Database Sizes

6.28.6.1 Use the Configurator

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 to change the size of the routing databases. This is because the configurator can perform checks that help to prevent a configuration that requires more memory than is available. In addition, the configurator checks, 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 the configurator (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.

6.28.6.2 How the DECNIS Configurator Manages Routing Database Sizes

The DECNIS 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 6–5. 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.

6.28.7 Choosing New Values for the Routing Databases

6.28.7.1 Total Memory Available

The total memory available for your configuration is 3500 Kbytes. 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 6–5 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.

6.28.7.2 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, issue the following command:

```
NCL> SHOW ROUTING ALL ATTRIBUTES
```

This will return the values of the status and characteristic attributes listed in Table 6–3.

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.

Use the configurator to make the adjustments required, and reboot the DECNIS.

You may need to repeat this process to fine-tune the sizes of the routing databases.

6.28.7.3 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.

6.28.8 Memory Allocation on the DECNIS

Table 6–5 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

The following table 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 below.

Table 6–5 DECNIS Memory Required for Each Configuration Parameter

Configuration Parameter	Minimum Value	Default	DECNIS Bytes Required for Each Unit Increment
Total memory available = 3500 Kbytes			
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	10	400[1]	120
L2 LSPs	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	–

[1]This is the value that the DECNIS will calculate if the other values in this table are the defaults.

[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 6–5 (Cont.) DECNIS Memory Required for Each Configuration Parameter

Configuration Parameter	Minimum Value	Default	DECNIS Bytes Required for Each Unit Increment
IP L2 connectivity factor	10	20	–
OSPF Maximum Connected Areas	1	2	19,264
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

[3] You can have up to 250 NetWare IPX circuits and 250 NetWare IPX services.

(continued on next page)

Table 6–5 (Cont.) DECNIS Memory Required for Each Configuration Parameter

Configuration Parameter	Minimum Value	Default	DECNIS Bytes Required for Each Unit Increment
NetWare IPX services[3]	0	0	100 + 40 per service per server
Circuits with NetWare IPX[4]	0	0	280
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

[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.

[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.

7
IP Routing

7.1 Introduction

This chapter describes how to manage IP routing on the DECNIS. It describes the following protocols:

- Integrated IS–IS
- OSPF
- RIP
- EGP

In addition, it describes the use of SNMP to monitor the IP information on the DECNIS.

7.1.1 NCL Commands

All tasks performed using NCL commands assume that you have already logged on to a suitable host system on the network and started NCL, and have set default to the DECNIS as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

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.

7.2 Using the DECNIS as an IP Router

7.2.1 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 6.3)
- FDDI (see Section 6.4)
- HDLC (see Section 6.5)
- DDCMP (see Section 6.5)
- PPP (see Section 6.7)
- CHDLC (see Section 5.18)
- VCP (see Section 5.8)
- X25 DA (see Section 6.15)
- X25 STATIC OUTGOING (see Section 6.16)
- X25 STATIC INCOMING (see Section 6.17)
- X25 PERMANENT (see Section 6.18)

The same circuit can be used for both IP routing and DECnet/OSI routing.

7.2.2 Example IP Network

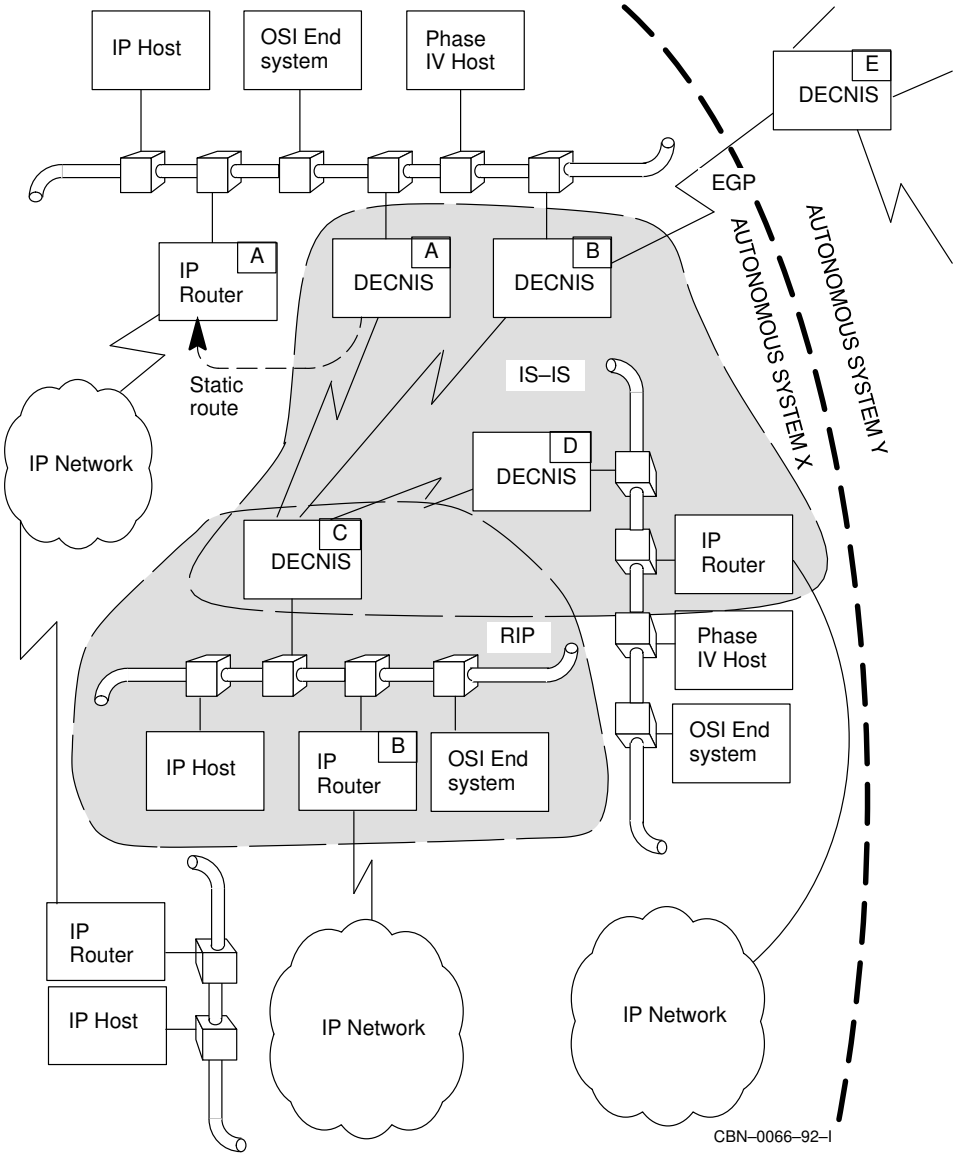
Figure 7–1 shows an example IP network using DECNIS systems as IP routers.

7.2.3 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 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.

Figure 7–1 Example IP Configuration



- On HDLC, DDCMP, and X.25 static routing circuits, the DECNIS can only forward IP packets to other Digital IP routers.

7.2.4 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.

7.3 IP Protocols

7.3.1 Protocols Supported by DECNIS

The DECNIS supports the following IP routing protocols:

- Interior Gateway Protocols:
  - Integrated IS–IS (defined in RFC 1195)
  - OSPF (defined in RFC 1247)
  - RIP (defined in RFC 1058)
- Exterior Gateway Protocol:
  - EGP (defined in RFCs 827, 888, and 904)

7.3.2 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.

7.4 IP Addressing

7.4.1 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.

7.4.2 Subnet Masks

Each IP address has an associated subnet mask. This is also a 32-bit binary number, and identifies the network address and subnetwork address parts of the IP address. A subnet mask consists of contiguous 1s followed by contiguous 0s. The contiguous 1s indicate the subnet address field 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.

7.4.3 Example

Consider an IP address of 24.45.21.8. This could have an associated subnet mask of 255.255.0.0, that is, all binary 1s in the first two octets. This indicates that the IP address is part of an IP subnetwork in which all IP addresses are of the form 24.45.*x.y*. The variable *x.y* identifies the host field of the IP address, and is different for each IP address. In this example, the host field is 21.8.

7.4.4 Further Reading

The *DECnet/OSI Routing Overview* describes IP addresses, subnetworks, and subnet masks.

7.5 Assigning IP Addresses and Subnet Masks

7.5.1 Subnets

If an IP network to which the DECNIS is connected is divided into subnets and you are running RIP in this network, 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. If, however, you are running OSPF, you can implement user-definable 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.

7.5.2 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 7.19.

7.5.3 LAN Routing Circuits

You must assign a unique IP subnet address to each DECNIS LAN routing circuit over which you want to run IP.

7.5.4 Point-to-Point Routing Circuits

If you intend to run RIP or EGP over the point-to-point circuit, you must supply either an IP subnet address for the circuit, or a Neighbor IP address. Section 7.9.2 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 Default IP address. Section 7.6 describes how the Default IP address is set.

Note, special rules apply to PPP routing circuits, see Section 7.5.5.

7.5.5 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.

This means that 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.

  The System IP Address status attribute is derived from the Manual IP Address. Section 7.6.2 describes how to set the Manual IP Address. If, for example, 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.

The address that is advertised by PPP is set in the Negotiated Neighbor IP Address characteristic of the Routing Circuit attribute on the remote system.

Note that, 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
```

7.5.6 Procedure for Assigning IP Subnet Addresses

Assign IP addresses as follows.

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
   ```

7.6 Default IP Address

7.6.1 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 circuit over which the packet is transmitted. If you have not assigned an IP address to the routing circuit from which the packet is transmitted, the source address contained in the packet will be the default IP address for the DECNIS.

You must set up a default IP address. Section 7.6.2 describes the command required to set the default IP address.

7.6.2 Setting a Default IP Address

Set a default 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.

7.7 Using the Integrated IS–IS Protocol for IP Routing

7.7.1 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 that you do not wish to use IS–IS for IP routing.

7.7.2 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 6.22), 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 6.22), 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.

7.7.3 Procedure

If you want to use the IS–IS protocol for IP routing, specify this in the configurator. 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.

7.8 Configuring the DECNIS to Use the OSPF Protocol

7.8.1 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.

7.8.2 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.

7.8.3 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.

7.8.4 OSPF Interface Types

An OSPF interface is the connection between a router and a network, or between two routers.

Note

'Interface' is OSPF terminology and 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. Neighboring routers are also discovered on these networks 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.

7.8.5 Router Classification

Within an autonomous system, routers can be split into OSPF areas. Figure 7–2 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.

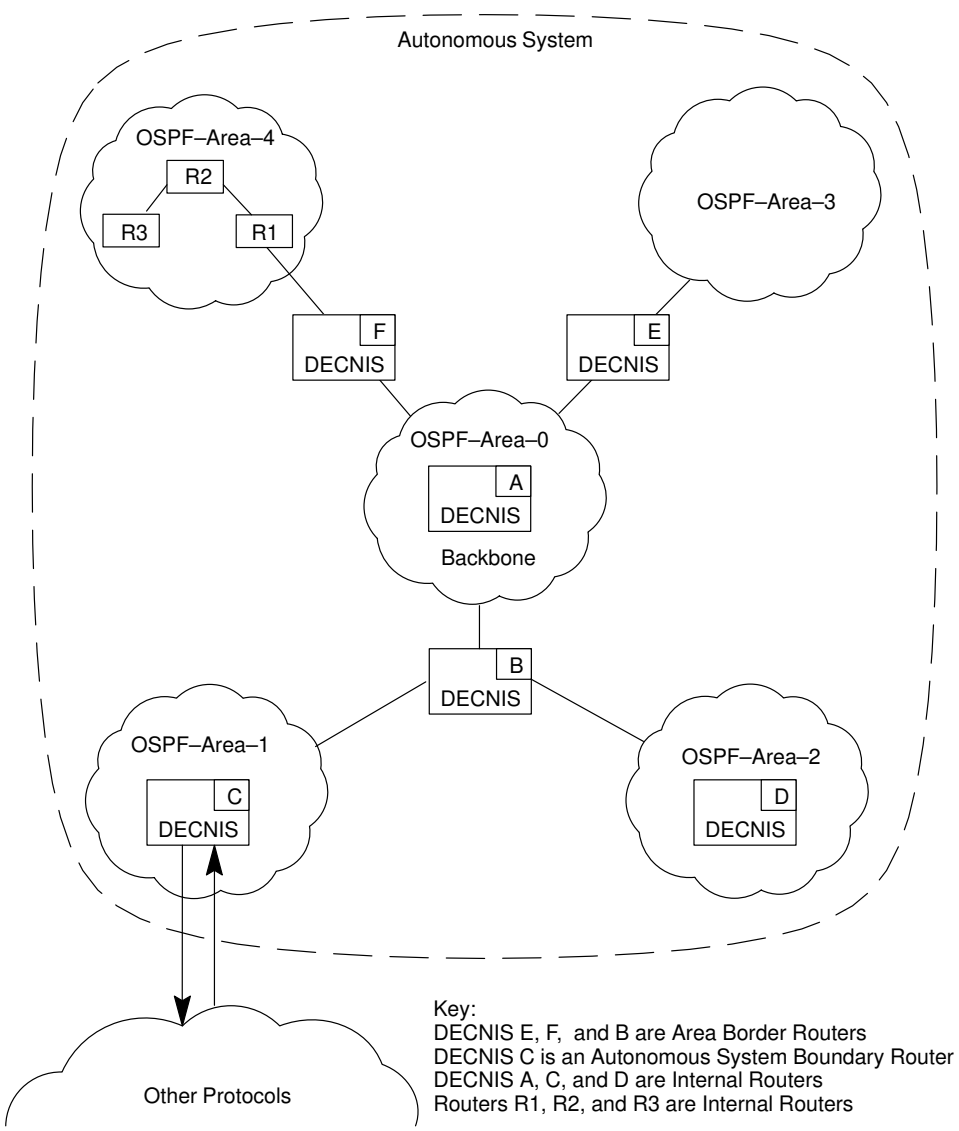
- Area Border Router

  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.

Figure 7–2 Different Functions of an OSPF Router



- 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.

7.8.6 OSPF CONTROL PROTOCOL Entities

In DECNIS, OSPF is managed by the CONTROL PROTOCOL entities, which are subentities of Routing.

There are two types of OSPF CONTROL PROTOCOL entities:

- OSPF GENERAL

  One OSPF GENERAL CONTROL PROTOCOL entity is needed per router. This defines the OSPF instance and contains the database sizing parameters. It can be set up by the Configurator (see Section 7.8.6.1).

- OSPF AREA

  One OSPF AREA CONTROL PROTOCOL entity is needed per area. This defines an OSPF area and is the parent to which the interfaces (logical circuits) are attached. OSPF AREA CONTROL PROTOCOL entities are set up outside of the Configurator (see Section 7.8.6.2).

7.8.6.1 Using the DECNIS Configurator to Set Up OSPF

The DECNIS configurator does not fully support OSPF. However, you can use the DECNIS configurator to set up the OSPF GENERAL CONTROL PROTOCOL entity. If you do, then you are automatically asked if you want the DECNIS to become an Autonomous System Boundary Router.

Creating the OSPF GENERAL CONTROL PROTOCOL Entity

In the Configurator Routing Section:

- 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
```

You will now be asked if you want the DECNIS to be an Autonomous System Boundary Router.

Setting Up the DECNIS as an Autonomous System Boundary Router

After selecting OSPF as an IP protocol in the Configurator Routing Section, you will be asked if you want to set up this DECNIS as an Autonomous System Boundary Router:

- Select YES if you want this DECNIS to be an Autonomous System Boundary Router.

The following line is added to the master NCL script file:

```
SET ROUTING CONTROL PROTOCOL OSPF-General OSPF AUTONOMOUS SYSTEM -
  BOUNDARY ROUTER TRUE
```

7.8.6.2 Creating OSPF AREA CONTROL PROTOCOL Entities

The following examples show how to set up the OSPF AREA CONTROL PROTOCOL entities 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 1.7.

7.8.6.3 Example 1: Setting Up a Minimum OSPF Configuration

Figure 7–3 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

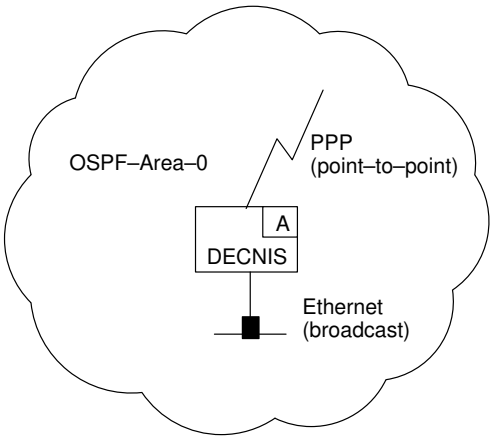
Follow these steps:

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 routing circuits with which they are associated.

Figure 7–3 OSPF Minimum Configuration




```
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 on the interface if you do not want to use the default priority of 1.

   In this example, DECNIS A is not eligible to become the Designated Router for the network attached to interface l601-3-0. Values 1 (lowest)—255 (highest) for OSPF PRIORITY mean that the router is eligible to become the Designated Router. A value of 0 means that it cannot become the Designated Router:

```
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
```

7.8.6.4 Example 2: Setting Up an OSPF Multiple Areas Configuration

Figure 7–4 shows a configuration with multiple areas. DECNIS B is an OSPF Area Border Router. It attaches areas 1.1.1.1 and 2.2.2.2 to backbone area 0.0.0.0, using three 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

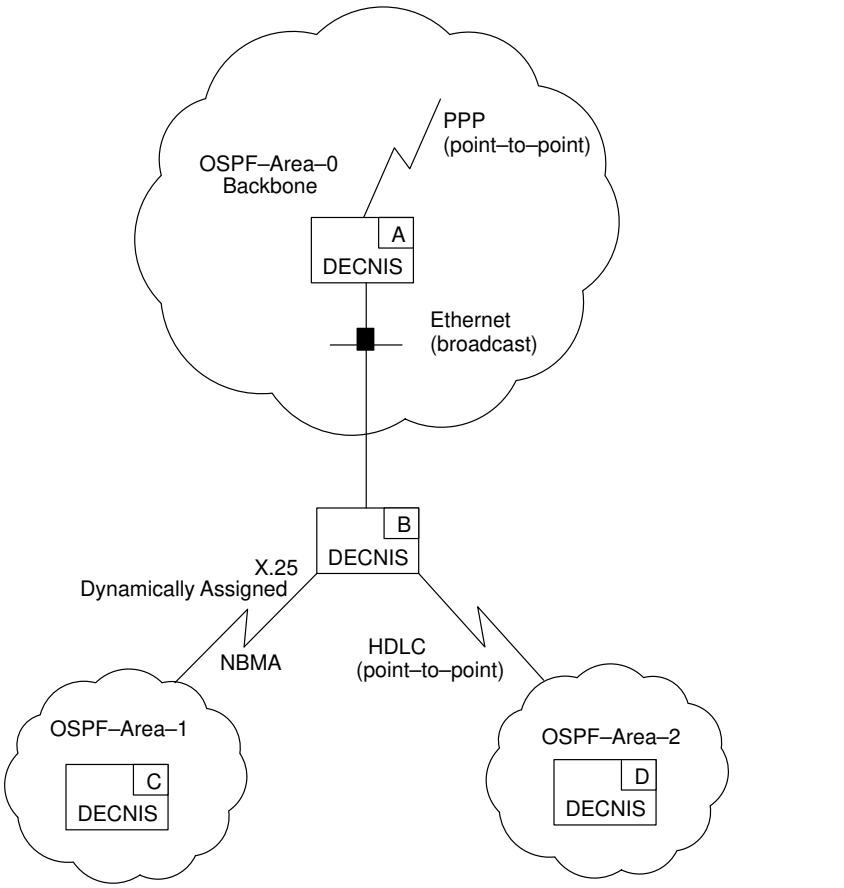
Follow these steps:

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
```

2. Create the OSPF interfaces (logical circuits) and give them the same names as the routing circuits with which they are associated.

Figure 7–4 OSPF Multiple Area Configuration




```
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
```

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 7.8.7 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
```

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
```

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
```

7. Enable the OSPF-General CONTROL PROTOCOL entity:

```
NCL> ENABLE ROUTING 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-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
```

7.8.6.5 Example 3: Connecting the OSPF Backbone Using Virtual Links

Figure 7–5 shows a virtual link configuration. A vitual link is used to connect two disjointed sections of the backbone area to make the backbone fully connected.

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.

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
   ```

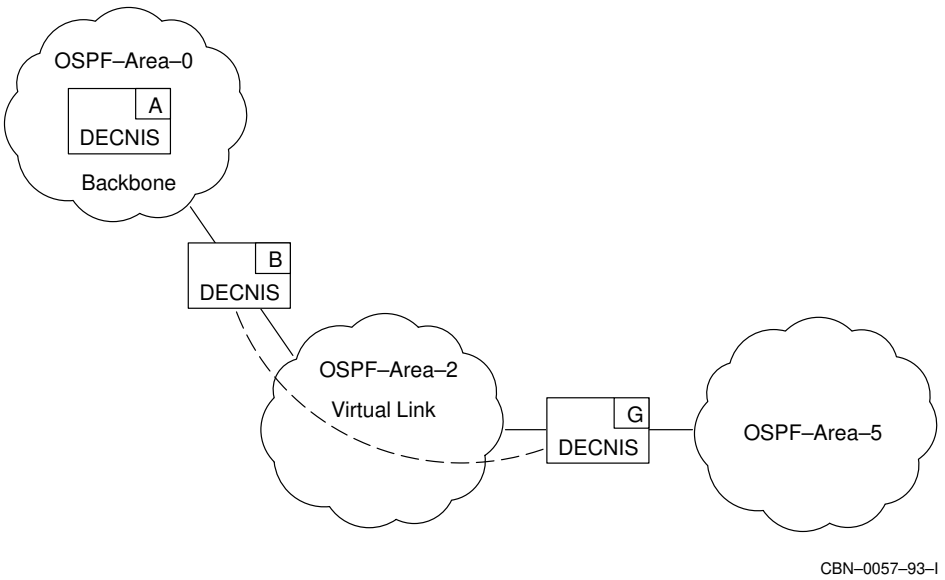
2. Set the Router ID of the virtual neighbor (DECNIS G, Router ID: 16.36.16.180) and the Transit Area (OSPF-Area-2):

   ```
   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
   ```

Figure 7–5 Using a Virtual Link to Connect to the OSPF Backbone



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
```

7.8.6.6 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
```

7.8.7 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 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 2 eligible Designated Routers so that one is elected Designated Router, and one is elected Backup Designated Router.

7.8.8 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.

7.8.9 Alternate Subnetworks

You can specify more than one IP address on a routing circuit. These are called alternate subnets.

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.

7.8.10 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

7.9 Configuring the DECNIS to Use the RIP Protocol

7.9.1 RIP Instances

The DECNIS allows you to create more than one instance of the RIP protocol. This allows you to control the reachability between subnets running RIP.

Figure 7–6 illustrates an example of how RIP can be set up. In the example, 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 7–6, 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 7–6, 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 8.5.

7.9.2 RIP Over Point-to-Point Circuits

If you want 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. To set a Neighbor IP Address, 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, do 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 7.5.5 for details about IP addressing.

7.9.3 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 7.9.2).
2. Create a CONTROL PROTOCOL entity for each RIP instance that you want to create. Enter the following command:

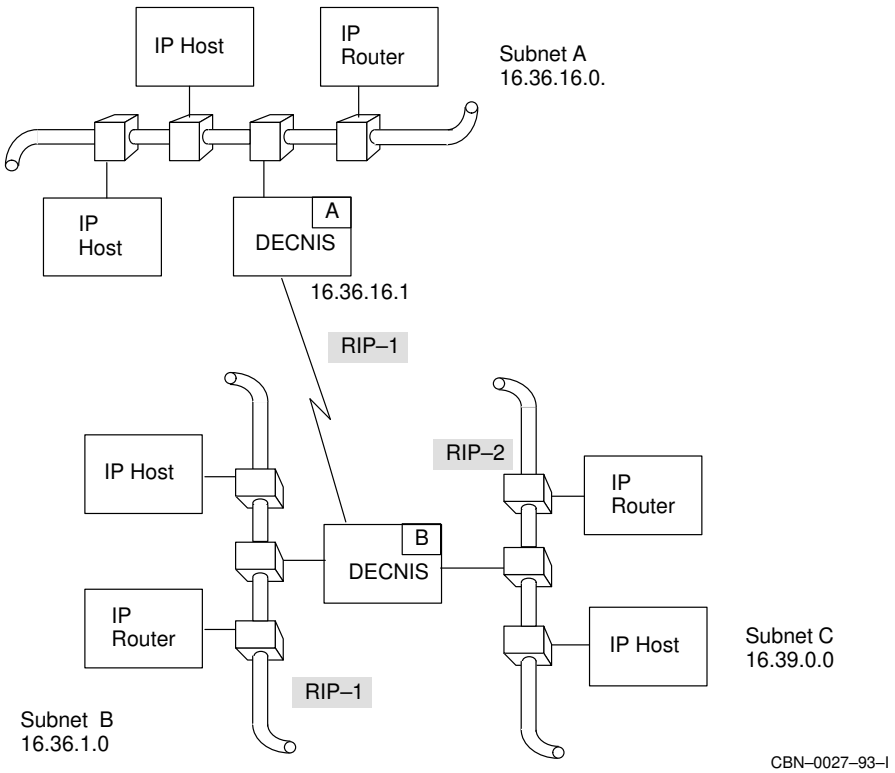
   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-protocol-name TYPE IP RIP
   ```

   where *control-protocol-name* is the name you want to give to this instance of RIP.

   For example:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL RIP-1 TYPE IP RIP
   ```

Figure 7–6 Using RIP in an IP Network



3. Associate each RIP instance with one or more subnets by entering the following command:

```
NCL> SET ROUTING CONTROL PROTOCOL control-protocol-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 mask of the subnet address.

  For example, in Figure 7–6, 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 7–6, 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 7.5.5 to determine what address is required.

4. Specify whether RIP messages of this instance are to be sent or received or both. Enter the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-protocol-name RIP STATE state
   ```

   where *state* is either:

   - 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-protocol-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-protocol-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-protocol-name RIP MODE mode
   ```

   where *mode* is either:

   - 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-protocol-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 as described above, 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 have 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-protocol-name
   ```

7.9.4 Using the Configurator to Set Up RIP

You can set up the RIP protocol using the DECNIS 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 DECNIS configurator sets up names and default values as shown in Table 7–1.

Table 7–1 RIP Configurator Defaults

Rip State	Receive Only, Send and Receive, Send Only
Control-protocol-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 determine how RIP has been set up, look at the master NCL script that is created when you exit the configurator.

If you want to change any of the RIP settings, you must edit the user NCL script files and reboot the DECNIS. For details about the user NCL script files, refer to Section 1.7.

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}}
```

7.10 The EGP Protocol

7.10.1 Example of Using the EGP Protocol

The EGP protocol is used to exchange IP routes between autonomous systems. Figure 7–7 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 the EGP protocol.

Note that, in Figure 7–7, 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.

7.10.2 Routing Level Required to Run EGP

The DECNIS must be configured as a Level 2 router in order to run EGP.

7.11 Configuring the DECNIS to Use the EGP Protocol

7.11.1 Example

An example of a DECNIS running EGP is shown in Figure 7–8.

7.11.2 Procedure

Follow these steps to allow the DECNIS (DECNIS B in Figure 7–8) to run the EGP protocol:

1. Ensure that the Autonomous System Number of the DECNIS (DECNIS B in Figure 7–8) has been set:

   ```
   NCL> SHOW ROUTING AUTONOMOUS SYSTEM NUMBER
   ```

   This 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, the DDN Network Information Center in Chantilly, Virginia, USA.

   If the Autonomous System Number has not been set, you must set it using the configurator, and reboot the DECNIS.

Figure 7–7 Communication Among Autonomous Systems Using the EGP Protocol

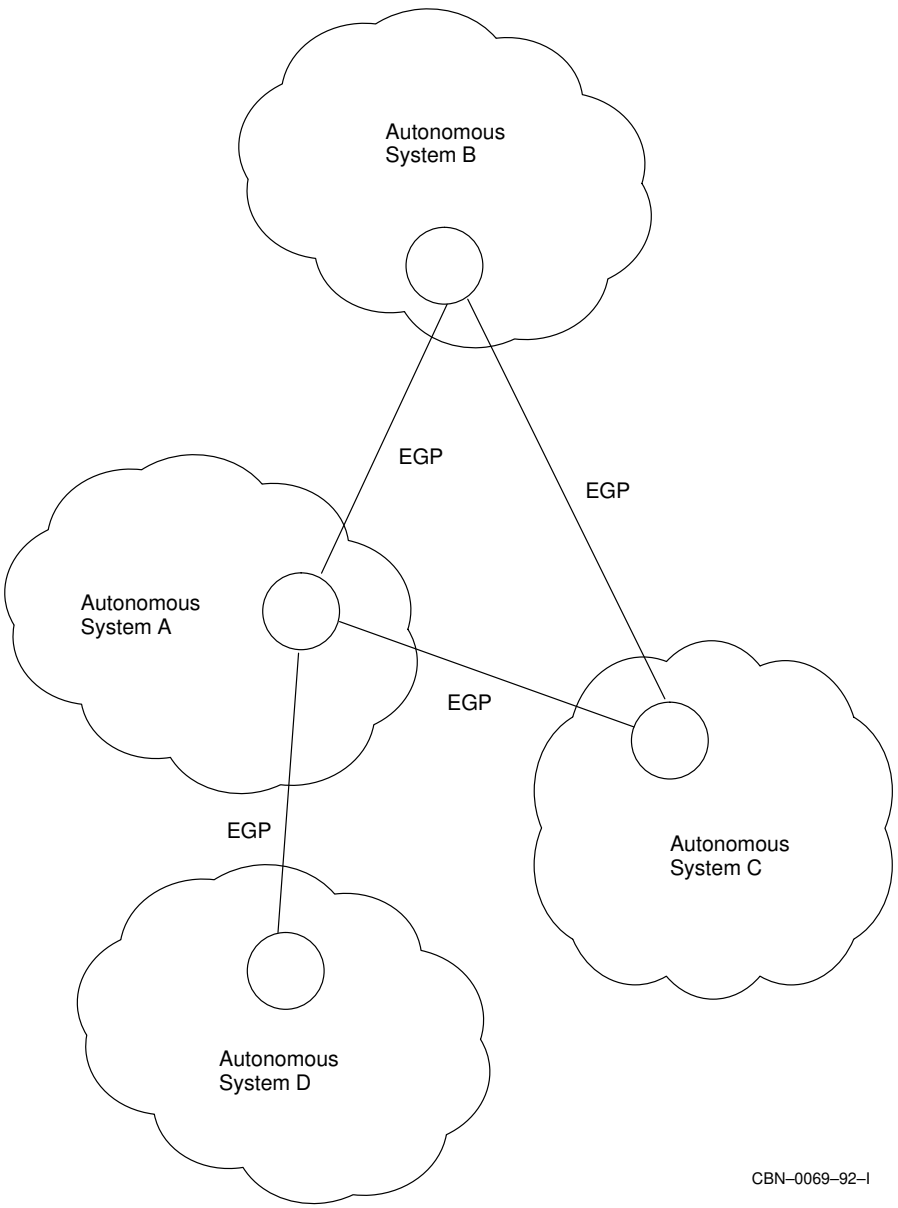
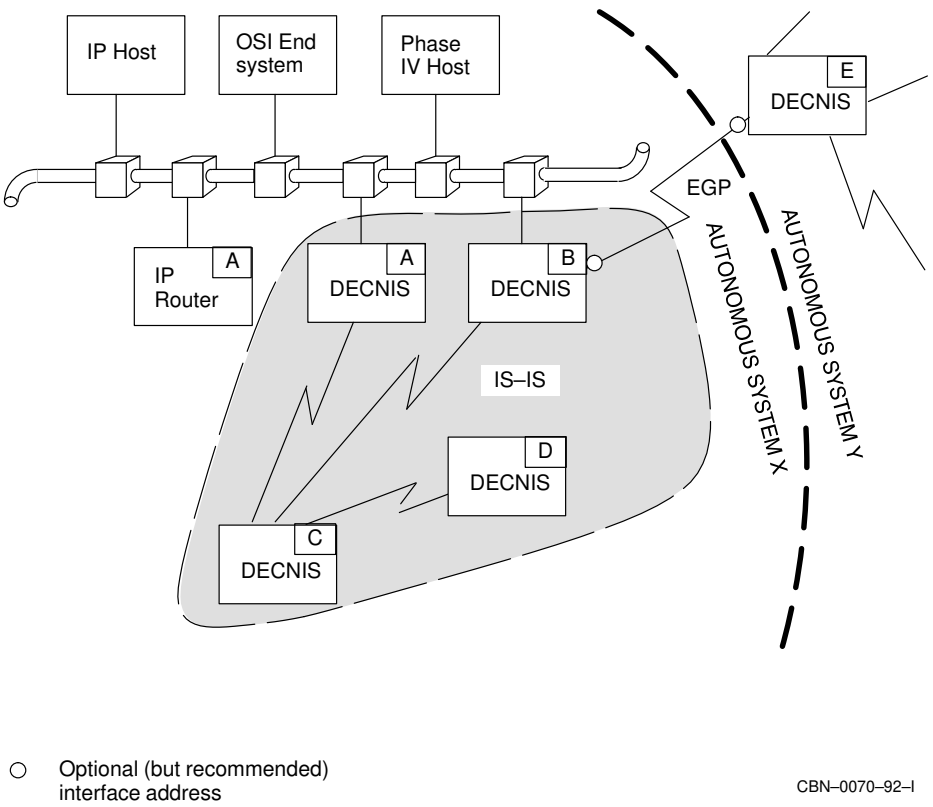


Figure 7–8 Using EGP in an IP Network



2. Assign an IP address and subnet mask to the EGP interface as shown in Figure 7–8: see Section 7.5.
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, the DDN Network Information Center in Chantilly, Virginia, USA.

   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
```

7.11.3 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 to. Each CONTROL PROTOCOL entity is known as an EGP instance.

Carry out the following steps to set up a CONTROL PROTOCOL entity:

1. Create the entity by entering the following command:

   ```
   NCL> CREATE ROUTING CONTROL PROTOCOL control-protocol-name TYPE EGP
   ```

   where *control-protocol-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 number of the Autonomous System Number for the EGP GROUP. For example, in Section 7.11.2, 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-protocol-name
   ```

For more information about route propagation and filtering, refer to Chapter 8.

7.11.4 Using the Configurator to Set Up EGP Propagation

You can set up the EGP protocol using the DECNIS configurator. If you specify that you want to propagate EGP routes, the 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 determine how EGP 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 EGP settings, you must edit the user NCL script files and reboot the DECNIS. For details about the user NCL script files, refer to Section 1.7.

7.12 Static Routes

7.12.1 Introduction

Static IP routes are similar to static OSI routes (see Section 9.14).

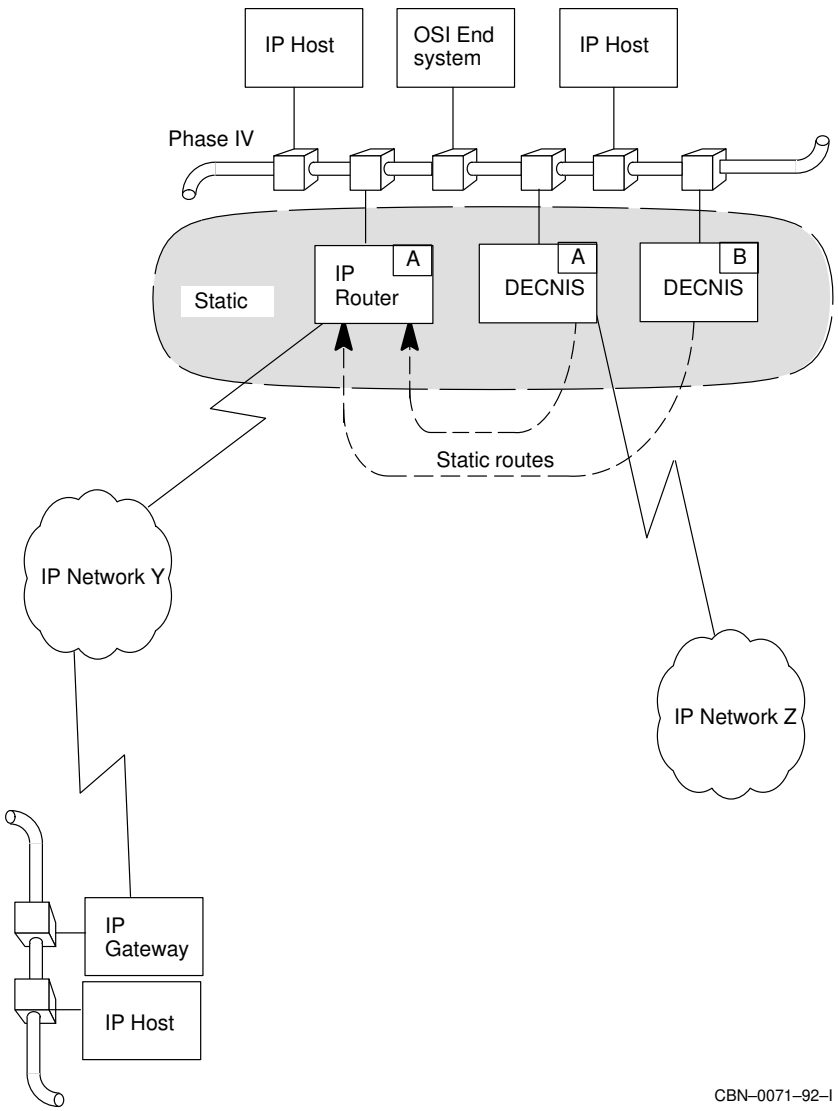
This section explains how to configure the DECNIS so that it can send IP packets to IP routers with which it does not exchange routing information, for example, if the circuit type is X25 DA.

7.12.2 Example

In the example shown in Figure 7–9, 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 7–9 Setting Up a Static Route



7.12.3 Setting Up a Static Route

Set up a static route on the DECNIS as follows:

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 7–9, *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 7–9), 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 7–9.

   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
   ```

7.13 Default Routes

7.13.1 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 database.

If an IP router receives a packet for a destination for which there is no entry in its routing database, it will forward the packet to an IP router that is advertising a default route.

7.13.2 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 database. If the DECNIS receives an IP packet for a destination not in its routing database, 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 database. If it then receives an IP packet for a destination not listed in its routing database, 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 database, it will send it to the adjacent system.

  Refer to Section 7.13.3 for details about 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 8.4. The DECNIS can generate a default route using route propagation; see Section 8.5.4 for details.

7.13.3 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 7.12 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
   ```

7.14 Abbreviating IP Routes in IS-IS Level 2 Routers

7.14.1 Introduction

By default, IP routing information is propagated throughout the autonomous system. However, by configuring your network so that each subnet is in a different DECnet/OSI area, and setting a summary address routing characteristic on each Level 2 router, only summary IP address information is sent between areas.

7.14.2 Example

Figure 7–10 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.

7.14.3 Summary IP Information and Routing Level

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.

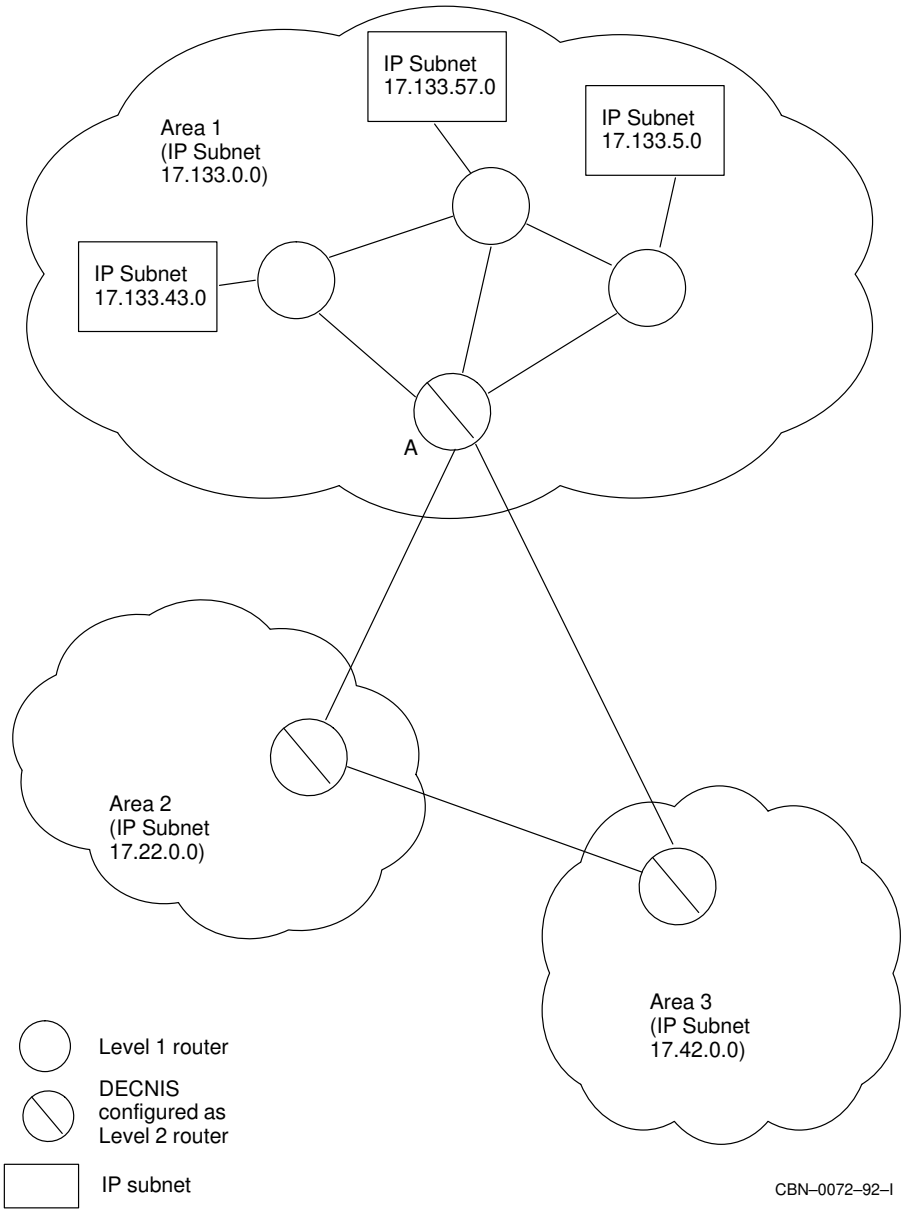
7.14.4 Setting Up Summary IS-IS Addresses on the DECNIS

On DECNIS A in Figure 7–10, 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 7–10 Using Abbreviated Routes



7.15 Setting Up Summary OSPF Addresses on the DECNIS

7.15.1 Introduction

OSPF Areas are identified to the IP network by their addresses, which 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.

7.16 Setting Up Summary Routes for an OSPF Area

You may want to define subnet masks to suit your configuration. The implementation of variable-length IP subnet masking allows flexibility in definition of address bits in the subnet mask. For example, you may have an OSPF network with many hosts and few subnetworks. If you thought it necessary, you could define subnet masks to identify more hosts than subnetworks, thus accommodating future growth of the network.

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.

7.16.1 Example

The following example shows how to configure Summary routes for an Area. The example refers to Figure 7–10.

1. Create a single Summary route for the networks in the range 17.133.0.0 to 17.133.255.255. Unlike the IS–IS case, 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.

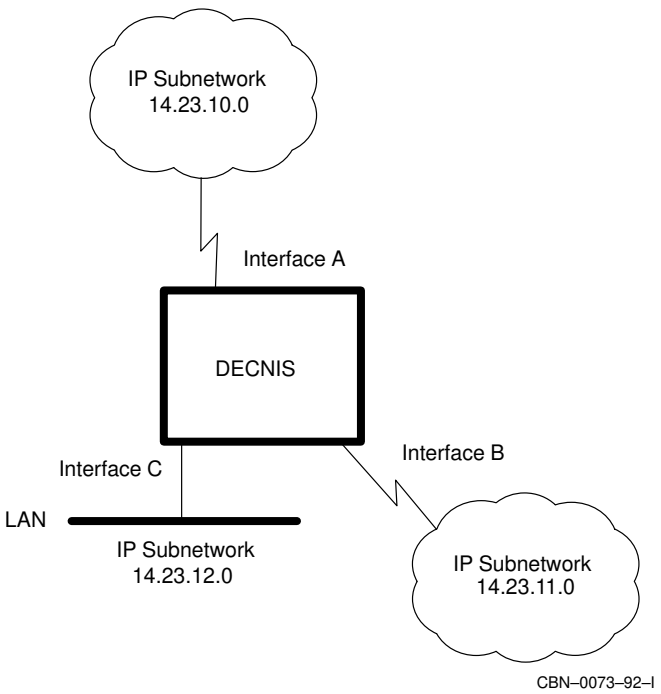
```
NCL> SET ROUTING CONTROL PROTOCOL OSPF-Area-1 -
_NCL> OSPF AREA RANGE { (ADDRESS = 17.133.0.0, MASK = 255.255.0.0) }
```

7.17 Using All Subnets Broadcast

7.17.1 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.

Figure 7–11 All Subnets Broadcast



7.17.2 Example

For example, consider Figure 7–11. 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. This is known as All Subnets Broadcast, and is described in detail in RFC 922.

Warning

You must ensure that your IP network is correctly configured before you enable All Subnets Broadcast. If there are any loops in the network, a large amount of excess traffic could be generated.

7.17.3 Data Link Broadcast

Note that the DECNIS will never forward a packet that it receives by data link broadcast. For example, in Figure 7–11, 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.

7.17.4 Procedure

Force an IP interface to forward broadcast packets by entering the following command:

```
NCL> SET ROUTING CIRCUIT circuit-name ALL SUBNETS BROADCAST ON
```

7.18 Using Proxy ARP

7.18.1 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.

7.18.2 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.

7.18.3 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 database 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.

7.18.4 Example

Figure 7–12 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.

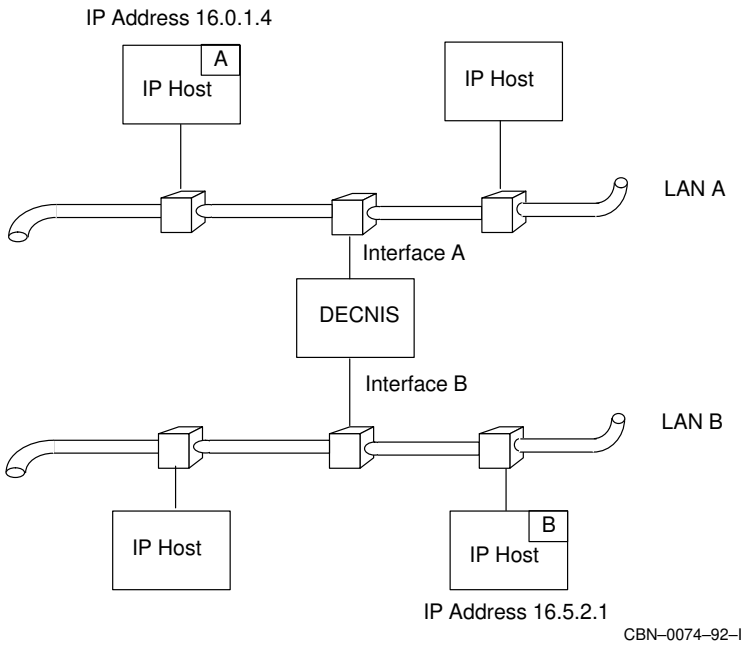
LAN B is an IP subnetwork, with an address of 16.5.0.0, and a subnet mask of 255.255.0.0.

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 7–12 DECNIS Using Proxy ARP



7.18.5 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 7–12, you must enter the command on Interface A and Interface B.

7.19 Using Alternative Subnet Addresses

7.19.1 Introduction

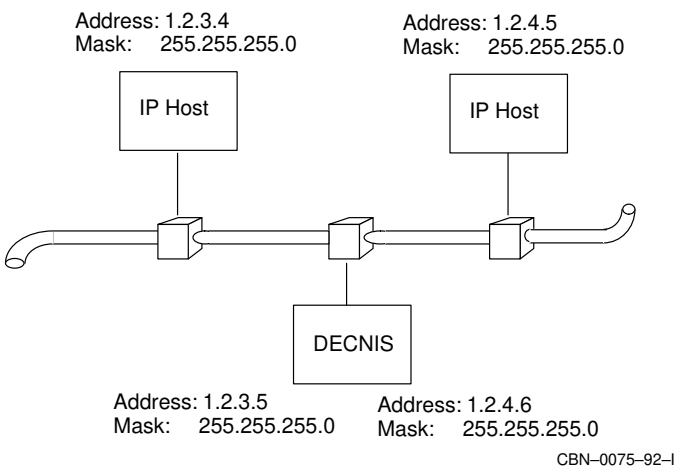
Normally, ARP messages are only sent between systems that are in the same subnet. If there are two or more subnets on the same LAN, the DECNIS can be configured to route messages between them, as shown in Figure 7–13.

To configure the DECNIS to route between different subnets on a LAN, you must give the DECNIS one or more alternative subnet addresses.

7.19.2 Example

In Figure 7–13, 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 7–13 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.

7.19.3 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
```

7.20 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 7.19) 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 in the forwarding table 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 7.11).

8

IP Route Propagation and Filtering

8.1 Introduction

This chapter describes the route propagation and filtering (RPF) functions for IP routing. These functions control how routes are received and advertised by the DECNIS.

8.1.1 NCL Commands

All tasks performed using NCL commands assume that you have already logged on to a suitable host system on the network and started NCL, and have set default to the DECNIS as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

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.

8.2 Introduction to Route Propagation and Filtering

The process for IP route filtering and propagation 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 route filtering

  By setting up route filtering, you can decide which routes are accepted into the Routing Table and how they are accepted. Refer to Section 8.4.1.4 for further information.

- Setting up route propagation

  By setting up route propagation, 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 8.5 for further information.

8.3 Entities Controlling Route Propagation and Filtering

Figure 8–1 shows the network management entities that control route propagation and filtering. Table 8–1 explains each component in the figure.

Figure 8–1 Entities that Control Route Propagation and Filtering

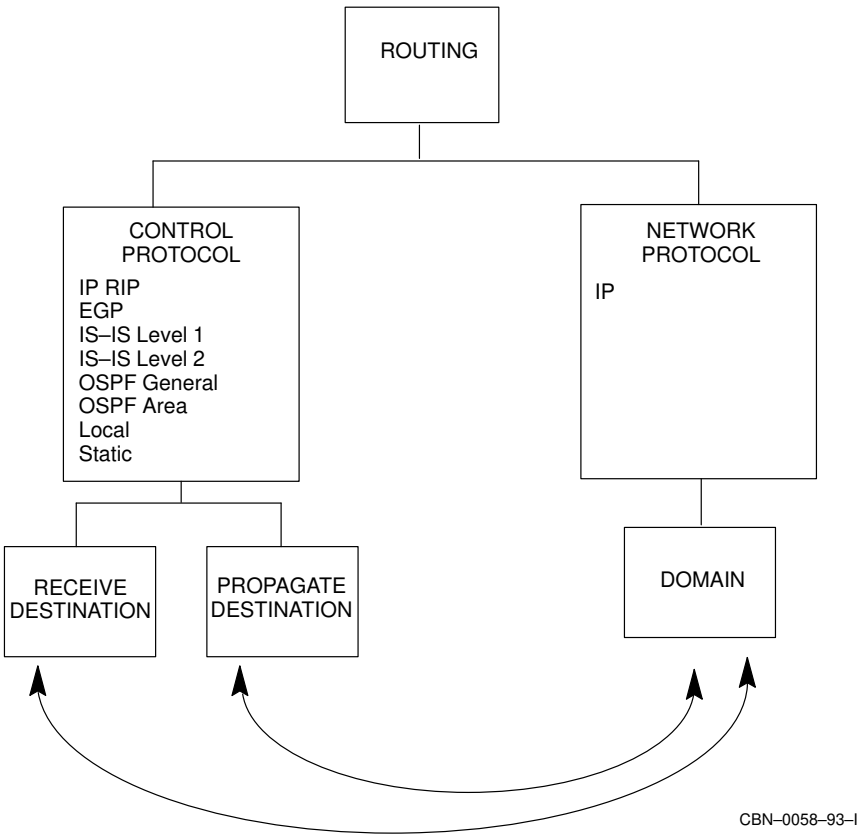


Table 8–1 Entity Descriptions

Entity	Description		
CONTROL PROTOCOL	This entity is a subentity of the Routing module. A CONTROL PROTOCOL entity is associated with each IP routing protocol in the DECNIS. The names of the entities are:		
	Route Type	**Route Type Control Protocol**	**Created**
	EGP	EGP	Manually when filtering EGP routes, or propagating into EGP. Refer to Section 7.11.3 and Section 7.11.4 to find out how to create and enable a CONTROL PROTOCOL entity for EGP.
	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	Any name	When setting up RIP.
	Local	LOCAL	Automatically when enabling Routing.
	Static	STATIC	Automatically when enabling Routing.
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	The NETWORK PROTOCOL entity is a subentity of the Routing module.		

(continued on next page)

Table 8–1 (Cont.) Entity Descriptions

Entity	Description
	For route filtering and route propagation, a NETWORK PROTOCOL entity of type IP must be created. This is created automatically when you set up IP using the DECNIS configurator.
DOMAIN	The DOMAIN entity 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 entities and the PROPAGATE DESTINATION entities may refer to the DOMAIN entity to determine which routes to filter and propagate. If you want to filter or propagate specific 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 and give them suitable names. 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.

8.4 Route Filtering

8.4.1 What Happens Without Route Filtering?

8.4.1.1 Advertisement Information Received

An example of route advertising is shown in Figure 8–2.

The information advertised by each neighboring router is listed in Table 8–2.

Figure 8–2 Route Advertising

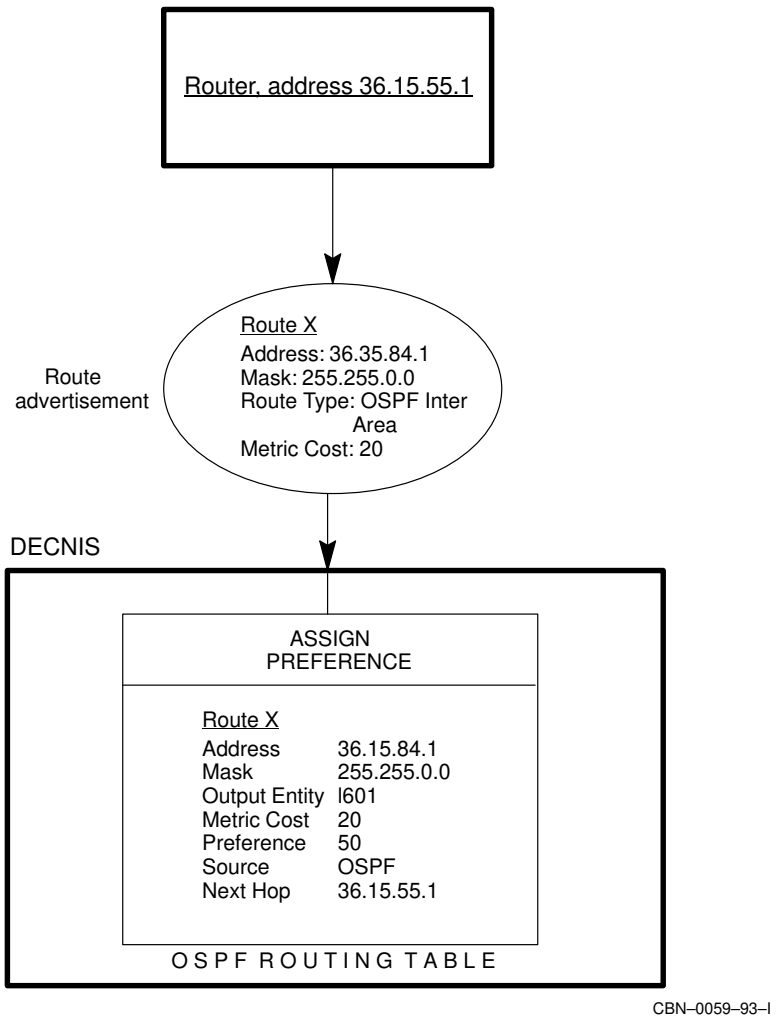


Table 8–2 Advertisement Information Received

Information	Explanation	
Address	The IP address.	
Mask	The IP address mask.	
Route Type	The route type which may take the following values:	
	Routing Protocol	**Route Types**
	EGP	EGP
	IS–IS	IS–IS Level 1
		IS–IS Level 2 Internal
		IS–IS Level 2 External
	OSPF	OSPF Intra-area
	OSPF	OSPF Interarea
	OSPF	OSPF External Type 1
	OSPF	OSPF External Type 2
	RIP	RIP
Metric Cost	The cost of this route to the destination address.	

Refer to Section 8.4.1.2 to find out what information is stored in the Routing Tables.

8.4.1.2 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 8–3.

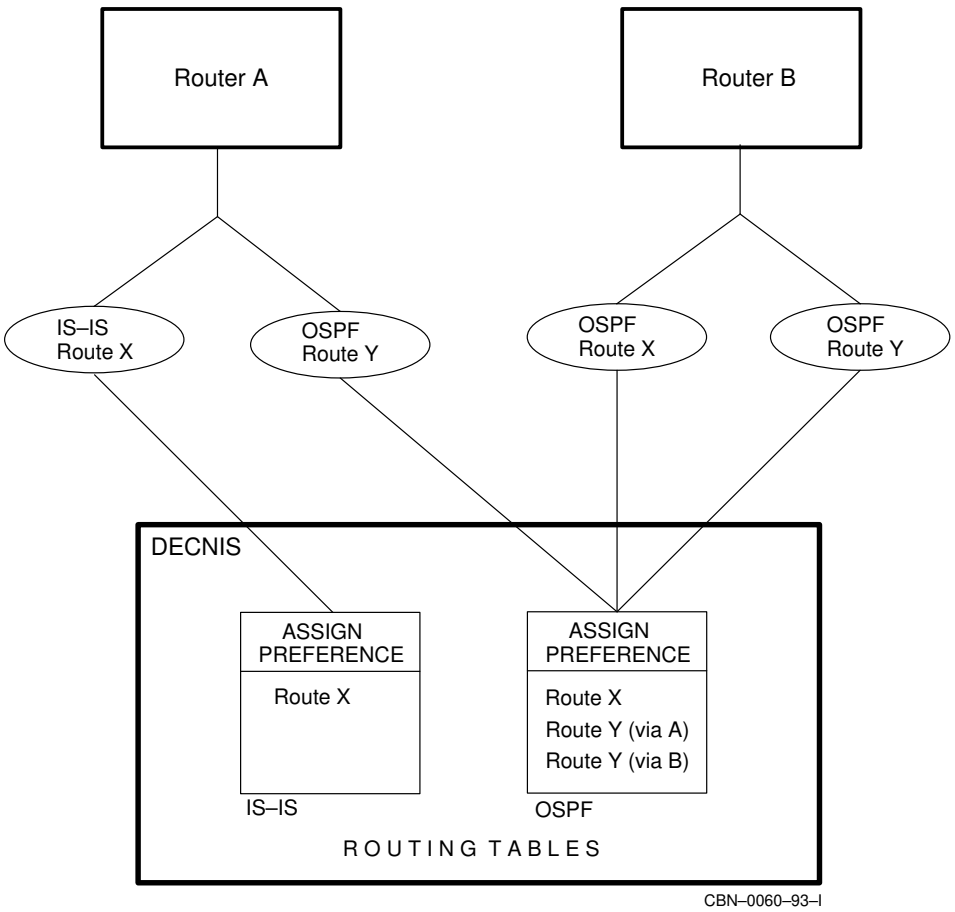
Table 8–3 Routing Table Data

Information	Explanation	
Address	IP address as received from the route advertisement.	
Mask	IP address mask as received from the route advertisement.	
Output Entity	The DECNIS interface through which data should be sent out to this address. Per definition this is the interface through which the route advertisement was received.	
Metric Cost	The metric cost of the route as received from the route advertisement.	
Preference	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. Default route preferences are as follows (from the most preferred to the least preferred):	
	Route Type	**Default Preference**
	Local	0
	Static	10
	IS–IS Level 1	20
	OSPF Intra-area	30
	IS–IS Level 2 Internal	40
	OSPF Inter Area	50
	OSPF External Type 1	60
	RIP	70
	EGP	80
	IS–IS Level 2 External	90
	OSPF External Type 2	100
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, in the case of OSPF a different, forwarding address may be given.	

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 8–3 shows how this can happen.

Figure 8–3 Multiple Occurrences of a Route



In Figure 8–3, 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.

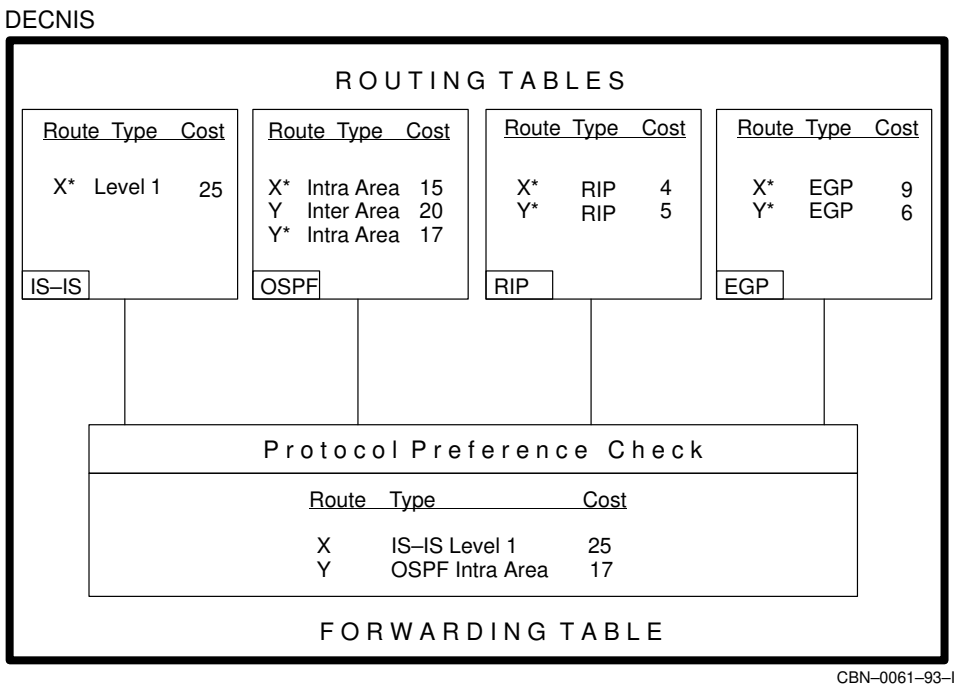
Route Y appears in the OSPF Routing Table twice because it was advertised by both Router A and Router B.

8.4.1.3 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 8–4.

Figure 8–4 Routes Passed to the Forwarding Table



In Figure 8–4, 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 8–3), and is therefore inserted into the Forwarding Table.

8.4.1.4 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. Filtering is used to achieve the following:

- To alter the metric cost of a route received

  This controls which routes are passed from each Routing Table to the Forwarding Table.

- To alter the default preference of a route received

This controls which route the Forwarding Table will choose as the 'most preferred' route.

- To block certain routes

  Routes may be blocked to prevent them from being inserted into the Routing Table altogether.

8.4.2 Setting Up Route Filtering

The steps involved in setting up route filtering are as follows:

1. Create a NETWORK PROTOCOL entity of IP. This has usually been done already. If not, 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. These are called IP domains. IP domains are set up under the NETWORK PROTOCOL entity so that they can be reused. You can create any number of domains. Refer to Section 8.9.1 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) you have set up in step 2. A RECEIVE DESTINATION FILTER is set up to:

   - Block routes
   - Pass routes but change their preference
   - Pass routes but change their metric cost

   Refer to Section 8.4.2.1 to find out how to set up a RECEIVE DESTINATION filter.

8.4.2.1 The RECEIVE DESTINATION Filter

Each RECEIVE DESTINATION has the characteristics shown in Table 8–4.

Table 8–4 RECEIVE DESTINATION Characteristics

For characteristic...	Specify...
DOMAINS	A list of IP domains that you have created 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.
METRIC	A new metric. Use in conjunction with a FILTER ACTION of PASS.

The characteristics DOMAINS, ADDRESS MATCH TYPE, and, in some cases, SOURCE ROUTE ATTRIBUTES, are used to check if the incoming route advertisement fulfills the criteria of the RECEIVE DESTINATION filter. If it does, then the FILTER ACTION, and/or any PREFERENCE or METRIC changes are applied to the route.

Figure 8–5 shows the route filtering process for CONTROL PROTOCOL 'A', which can be any IP control protocol instance.

8.4.2.2 Setting Up a RECEIVE DESTINATION Filter to Block Routes

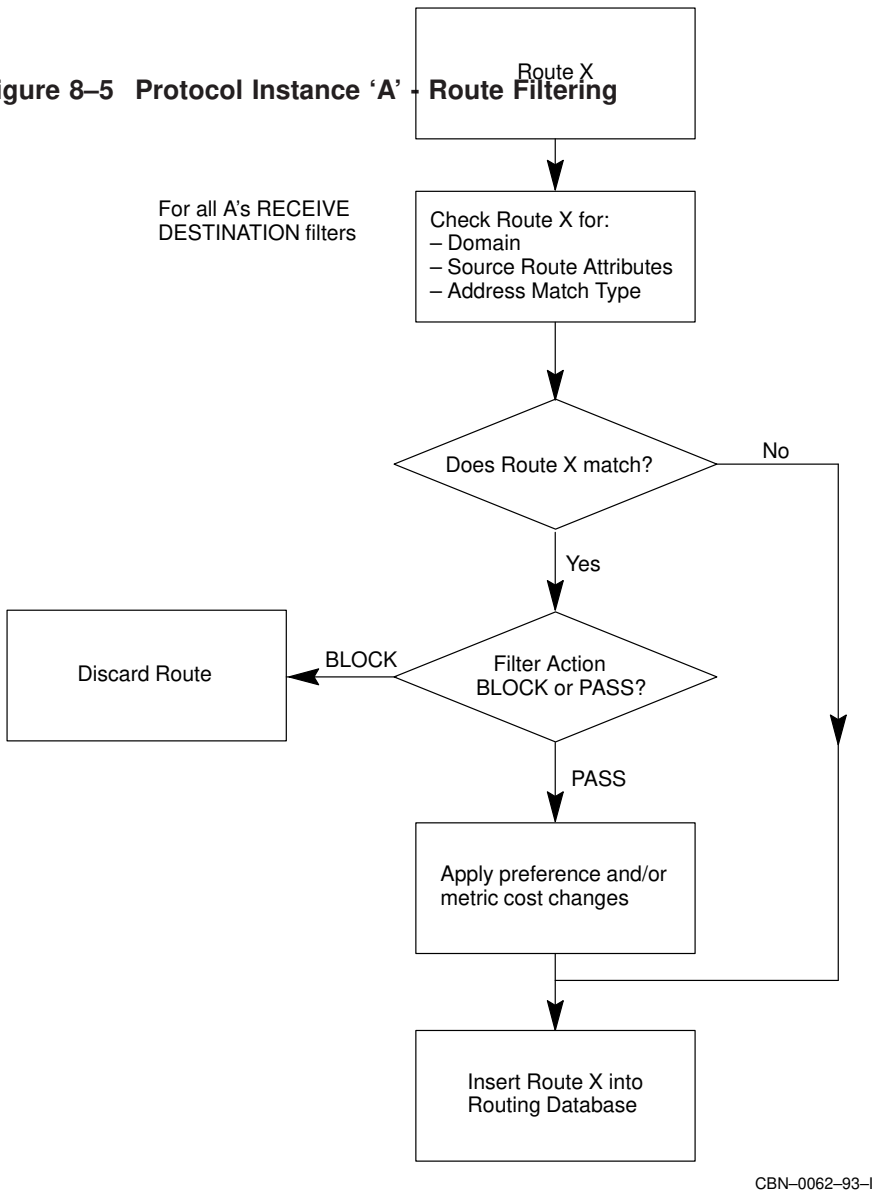
This section assumes that there is a ROUTING CONTROL PROTOCOL entity set up for the type of route you want to filter. For more details about this entity, see Section 8.3.

To set up the filter, carry out the following steps:

1. Create an IP domain specifying the routes that you want to block from being inserted into the Routing Table. Refer to Section 8.9.1 to find out how to create an IP domain. If you do not specify a domain, then all routes of that particular protocol will be blocked.
2. Create the filter by entering the following command:

```
NCL> CREATE ROUTING CONTROL PROTOCOL control-protocol-name -
_NCL> RECEIVE DESTINATION filter-name FILTER ACTION action, [SOURCE -
_NCL> ROUTE ATTRIBUTES (s-r-attribute)]
```

Figure 8–5 Protocol Instance 'A' - Route Filtering



where:

control-protocol-name is the name of the control protocol instance.

filter-name is the name of the receive filter.

action is either BLOCK or PASS.

s-r-attribute only applies when filtering the route types listed in Table 8–5.

For example:

```
NCL> CREATE ROUTING CONTROL PROTOCOL IS-ISLEVEL2 -
_NCL> RECEIVE DESTINATION FILTER_1 FILTER ACTION BLOCK, SOURCE -
_NCL> ROUTE ATTRIBUTES (ISIS INTERNAL)
```

Table 8–5 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. Now specify the IP domain(s) containing 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-protocol-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 8.9.1 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 8.4.2.6 and Section 8.4.2.7, respectively).

4. Specify whether the address matching is to be EXACT or PREFIX by entering the following command:

```
NCL> SET ROUTING CONTROL PROTOCOL control-protocol-name -
_NCL> RECEIVE DESTINATION filter-name ADDRESS MATCH TYPE type
```

   where *type* is EXACT or PREFIX.

   For details about this attribute, see Section 8.9.2.

5. Now enable the receive filter by entering the following command:

```
NCL> ENABLE ROUTING CONTROL PROTOCOL control-protocol-name -
_NCL> RECEIVE DESTINATION filter-name
```

Note

You cannot set up OSPF or IS–IS filters 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 1.7.

Refer to Section 8.7 for a description of setting up route filtering and propagation in an example IP network.

8.4.2.3 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 8–4). Refer to Table 8–3 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:

1. Create an IP domain specifying the routes that will be inserted into the Routing Table at higher preference. Refer to Section 8.9.1 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-protocol-name -
_NCL> RECEIVE DESTINATION filter-name FILTER ACTION PASS, -
_NCL> [SOURCE ROUTE ATTRIBUTES (s-r-attribute)]
```

where:

control-protocol-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 8–5.

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-protocol-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-protocol-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 ROUTER CONTROL PROTOCOL control-protocol-name -
_NCL> RECEIVE DESTINATION filter-name
```

For example:

```
NCL> ENABLE ROUTER 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 either IS–IS or OSPF. The preference of IS–IS and OSPF routes can only be changed if Routing is disabled.

8.4.2.4 Disappearing Routes

The preference system can appear 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.

Checking the Forwarding Table

You can check the routes in the Forwarding Table by entering the following command:

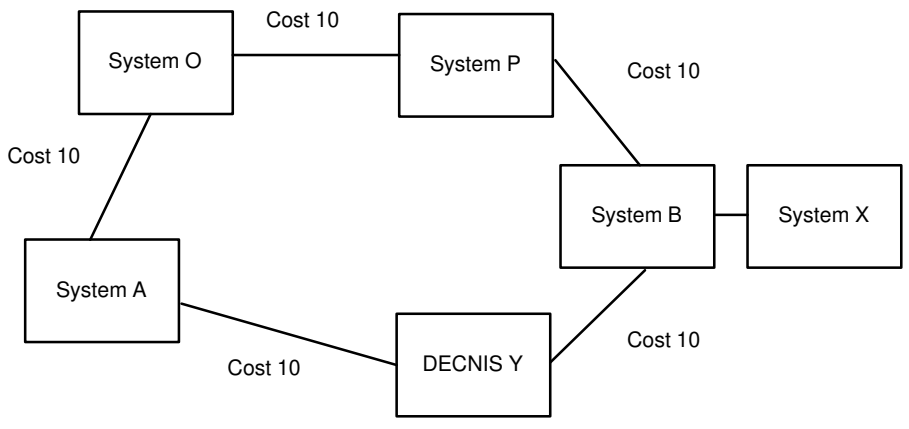
```
NCL> SHOW ROUTING IP DESTINATION ADDRESS * ALL
```

8.4.2.5 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 filtered out the route (the route does not exist in its routing database). Figure 8–6 shows an example of how a "black hole" can be created when using IS–IS filtering.

In Figure 8–6, 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 is filtered 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.

Figure 8–6 IS–IS/OSPF Filtering



8.4.2.6 Filtering All Routes of a Control Protocol

If you want to filter all the routes of a particular control protocol type:

- Do not specify the name of an IP domain.
- Set the address match type CHARACTERISTIC to PREFIX.

8.4.2.7 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 is filtered when the following situation exists:

– No IP domain is specified.

– The ADDRESS MATCH TYPE characteristic is set to EXACT.

8.5 Route Propagation

8.5.1 What Happens Without Route Propagation?

8.5.1.1 Route Advertising

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.

8.5.1.2 Advertising Routes Learnt by Another Routing Protocol

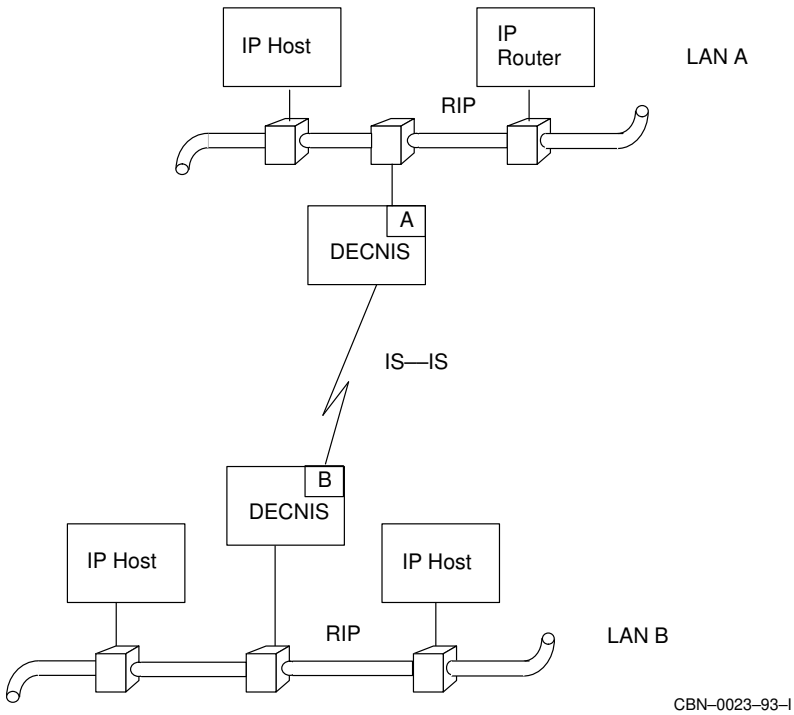
You may set up route propagation for any IP protocol instance to achieve the following:

- To advertise routes learnt by another IP routing protocol.

  For example, you can set up a filter for a RIP instance so that it advertises certain routes which were learnt by IS–IS.

For example, in Figure 8–7, 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 a filter for the relevant IS–IS protocol instance on DECNIS A which 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.

Figure 8–7 Route Propagation



For an example of RIP propagation, see Section 8.7.

For details about RIP instances and EGP instances, refer to Section 7.9 and Section 7.11.3.

8.5.2 Setting Up Route Propagation

The steps involved in setting up route propagation are as follows:

1. Create a NETWORK PROTOCOL entity of IP. This has usually been done already. If not, enter the following command:

   ```
   NCL> CREATE ROUTING NETWORK PROTOCOL IP
   ```

2. Set up the list of addresses you want to propagate. These are called IP domains. IP domains are set up under the NETWORK PROTOCOL entity so that they can be reused. You can create any number of domains. Refer to Section 8.9.1 to find out how to set up an IP domain.

3. Set up a PROPAGATE DESTINATION filter for the CONTROL PROTOCOL instance, using the IP domain(s) you have set up in step 2.

   Refer to Section 8.5.2.1 to find out how to set up a PROPAGATE DESTINATION filter.

Note

If you want to set up a PROPAGATE DESTINATION filter for OSPF, then the DECNIS must be set up as an Autonomous System Boundary Router.

8.5.2.1 The PROPAGATE DESTINATION Filter

Each PROPAGATE DESTINATION entity has seven characteristics:

Table 8–6 PROPAGATE DESTINATION Characteristics

For characteristic...	Specify...
SOURCES	The control protocols whose routes you want to propagate from. In order to match this PROPAGATE DESTINATION, the route in the forwarding database must come from the source specified in this characteristic.
DOMAINS	Any of the IP domains that you have created containing the addresses and masks of the routes you want to propagate.
ADDRESS MATCH TYPE	EXACT or PREFIX.
SOURCE ROUTE ATTRIBUTES	Specify only for IS–IS Level 2 routes and all OSPF routes. The type of route (for the control protocol specified in SOURCES) you want to propagate.
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 filters belonging to IS–IS Level 2 and OSPF protocol instances.

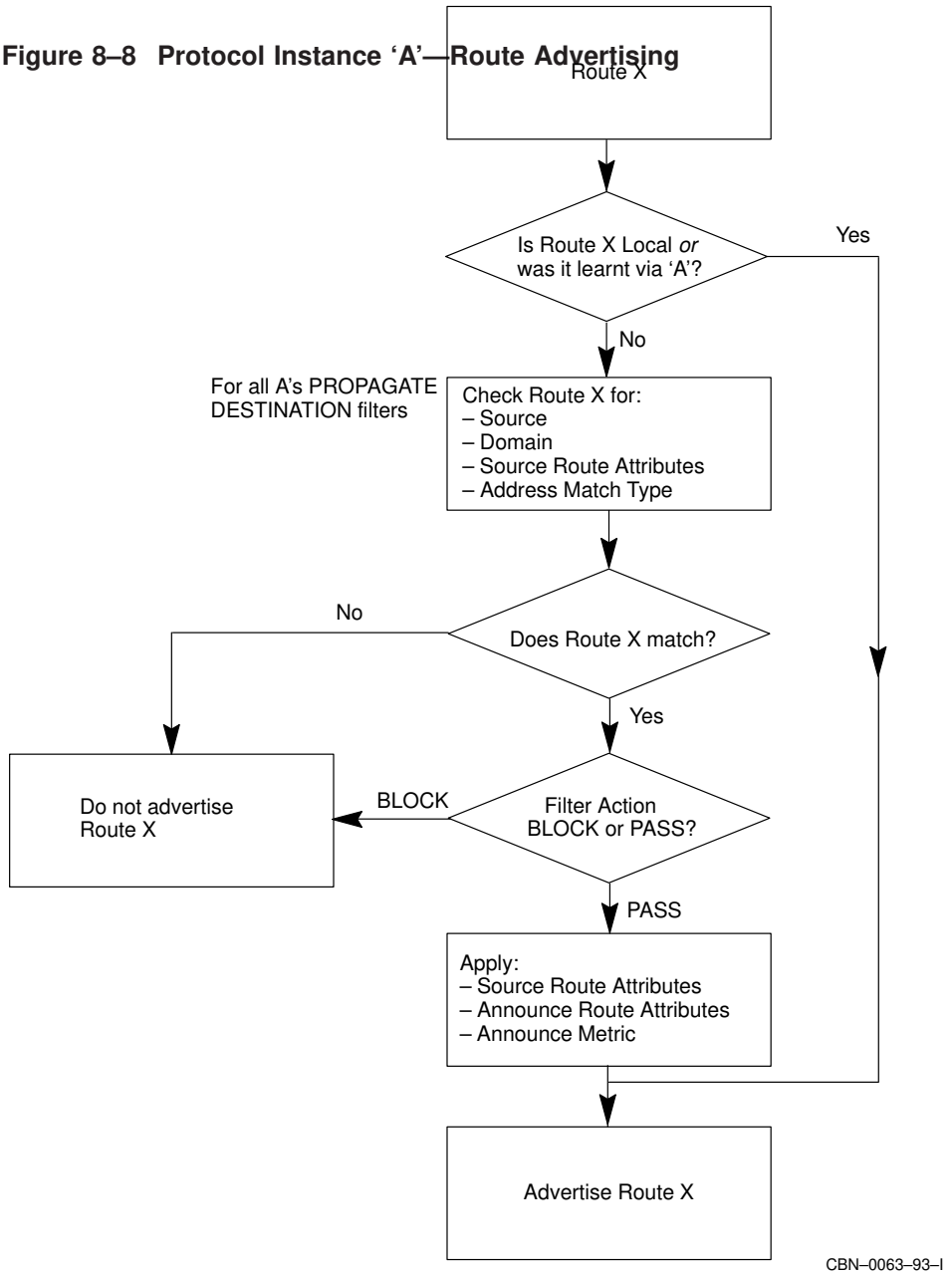
Together, these attributes form a PROPAGATE DESTINATION filter. The first four characteristics are used to check whether the route matches the PROPAGATE DESTINATION entity.

If the route in the Forwarding Table matches the characteristics of the PROPAGATE DESTINATION filter, it will be filtered 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 8–8 shows route propagation within the route advertising process for CONTROL PROTOCOL ‘A’ which can be any of the IP Routing CONTROL PROTOCOLs.

Any route that you want to be propagated must be in the Forwarding Table of the DECNIS.

Figure 8–8 Protocol Instance 'A'—Route Advertising



8.5.3 Setting Up a Filter to Propagate Routes

To set up route propagation on the DECNIS, carry out the following steps:

1. Create the NETWORK PROTOCOL IP entity.

   This should have been created when you set up IP routing. If it has not been created, enter the following command:

   ```
   NCL> CREATE ROUTING NETWORK PROTOCOL IP
   ```

2. Create an IP domain.

   Refer to Section 8.9.1 to find out how to set up an IP domain.

   Note

   For propagating all routes or the default route, do not set up an IP domain. For more details, see Section 8.9.2 and Section 8.5.4.

3. Create and set up a Propagate Destination filter.

   This section assumes that there is a ROUTING CONTROL PROTOCOL entity set up for the protocol type of the routes that you want to propagate from and for the protocol type you want to propagate into. For more details about the CONTROL PROTOCOL entity, see Table 8–1.

   To set up a PROPAGATE DESTINATION filter, carry out the following steps:

   1. Create the filter by entering the following command:

      ```
      NCL> CREATE ROUTING CONTROL PROTOCOL control-protocol-name -
      _NCL> PROPAGATE DESTINATION filter-name FILTER ACTION action, -
      _NCL> [SOURCE ROUTE ATTRIBUTES (s-r-attribute)]
      ```

      where:

control-protocol-name	is the protocol instance for which you are setting up this filter. You cannot set up a filter for an OSPF Area CONTROL PROTOCOL.
filter-name	is the name of the propagate 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 filter for a RIP protocol instance to advertise ISIS INTERNAL routes:

```
NCL> CREATE ROUTING CONTROL PROTOCOL RIP -
_NCL> PROPAGATE DESTINATION FILTER_1 FILTER ACTION PASS, -
_NCL> SOURCE ROUTE ATTRIBUTES (ISIS INTERNAL)
```

2. Specify the IP domain containing the routes you want to propagate. You can specify one IP domain or a set of IP domains. Enter the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-protocol-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 8.9.2 and Section 8.5.4.

3. Specify whether the address matching is to be EXACT or PREFIX by entering the following command:

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-protocol-name -
   _NCL> PROPAGATE DESTINATION filter-name ADDRESS MATCH TYPE type
   ```

   where *type* is EXACT or PREFIX.

   For details about this characteristic, see Section 8.9.2.

4. Specify the SOURCES characteristic.

   The SOURCES characteristic specifies the names of the control protocol instances that you want to propagate from. In other words, 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-protocol-name -
   _NCL> PROPAGATE DESTINATION filter-name SOURCES {{ROUTING CONTROL -
   _NCL> PROTOCOL control-protocol-name} ,...}
   ```

For example, if you want 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}, {ROUTING CONTROL PROTOCOL EGP}}
```

5. Specify the ANNOUNCE ROUTE ATTRIBUTES characteristic if this PROPAGATE DESTINATION filter is for an IS–IS or OSPF protocol instance. If this is the case, then you need to specify what type of route the propagated route needs to be advertised as.

   ```
   NCL> SET ROUTING CONTROL PROTOCOL control-protocol-name -
   _NCL> PROPAGATE DESTINATION filter-name ANNOUNCE ROUTE -
   _NCL> ATTRIBUTES (a-r-attribute)
   ```

   where you may set *a-r-attribute* to the following values:

Table 8–7 ANNOUNCE ROUTE Attributes

If you want the route to be advertised 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.

6. 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-protocol-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 8–8 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 database. To check the metric for the routes in the Forwarding database, 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 8–8.

7. Now enable the Propagate Destination filter by entering the following command:

```
NCL> ENABLE ROUTING CONTROL PROTOCOL control-protocol-name -
_NCL> PROPAGATE DESTINATION filter-name
```

Table 8–8 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

8.5.4 Advertising a Default Route

The DECNIS can be set up to advertise a default route using route propagation. Follow the procedures in Section 8.5.3 using the following settings:

- Do not specify an IP domain
- Set address matching to EXACT
- Set the SOURCES characteristic to LOCAL

8.5.5 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 8.5.3 using the following settings:

- Do not specify an IP domain
- Set address matching to PREFIX

8.5.6 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.

8.5.7 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).

8.6 Using the Configurator to Set Up Route Propagation

Note

There is no DECNIS configurator support for OSPF route propagation.

Except for OSPF, you can use the DECNIS configurator to set up a simple form of route propagation. 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 1.7.

If you want to propagate 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.

The following NCL shows an example NCL script 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 filters to define propagation of routes between protocols
!
! create filter 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 filter 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 1.7.

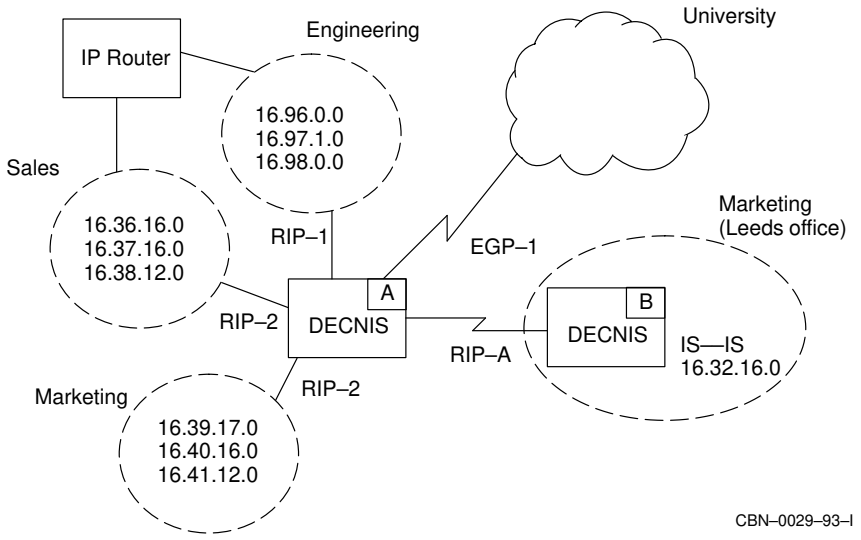
8.7 Examples of Setting Up Route Filtering and Propagation Using RIP, EGP, and IS–IS

Figure 8–9 shows an example IP network that might exist within a company. The following examples describe how you might use route filtering and propagation to control the flow of IP routing traffic through the network.

Note

Local routes cannot be filtered.

Figure 8–9 Example IP Network Using Route Filtering and Propagation



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 8–9, 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 7.9.
- 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
```

8.8 Examples of Setting Up Route Propagation Using RIP and OSPF

8.8.1 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 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 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 CONTROL PROTOCOL OSPF-General -
_NCL> PROPAGATE DESTINATION RIP1-pass
```

8.8.2 An Example of Propagating OSPF Routes into RIP

If you want 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 filters for RIP-1 and name the entities:

```
NCL> CREATE CONTROL PROTOCOL RIP-1 -
_NCL> PROPAGATE DESTINATION OSPF-Intra-pass -
_NCL> FILTER ACTION PASS, SOURCE ROUTE ATTRIBUTE { OSPF Intra Area }

NCL> CREATE CONTROL PROTOCOL RIP-1 -
_NCL> PROPAGATE DESTINATION OSPF-Inter-pass -
_NCL> FILTER ACTION PASS, SOURCE ROUTE ATTRIBUTE { OSPF Inter Area }

NCL> CREATE CONTROL PROTOCOL RIP-1 -
_NCL> PROPAGATE DESTINATION OSPF-Extyp1-pass -
_NCL> FILTER ACTION PASS, SOURCE ROUTE ATTRIBUTE { OSPF External Type 1 }

NCL> CREATE 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 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 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 CONTROL PROTOCOL RIP-1 -
_NCL> PROPAGATE DESTINATION OSPF-Extyp1-pass -
_NCL> SOURCE { ROUTING CONTROL PROTOCOL OSPF-General }, -
_NCL> ADDRESS MATCH TYPE PREFIX

NCL> SET 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 CONTROL PROTOCOL RIP-1 -
_NCL> PROPAGATE DESTINATION OSPF-Intra-pass

NCL> ENABLE CONTROL PROTOCOL RIP-1 -
_NCL> PROPAGATE DESTINATION OSPF-Inter-pass

NCL> ENABLE CONTROL PROTOCOL RIP-1 -
_NCL> PROPAGATE DESTINATION OSPF-Extyp1-pass

NCL> ENABLE CONTROL PROTOCOL RIP-1 -
_NCL> PROPAGATE DESTINATION OSPF-Extyp2-pass
```

8.9 IP Domains

8.9.1 Setting Up IP Domains

To set up an IP domain, enter the following command:

```
NCL> CREATE ROUTING NETWORK PROTOCOL IP DOMAIN domain-name
```

where *domain-name* is the name of the domain you want to create.

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}}
```

You can add addresses to an existing domain by entering 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} ,...}
```

Note

For filtering all routes or the default route, do not set up an IP domain.

8.9.2 IP Domains and the ADDRESS MATCH TYPE

This section describes how the setting of the ADDRESS MATCH TYPE affects which routes are filtered. The ADDRESS MATCH TYPE is set 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 EXACT. This means that 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.

If the ADDRESS MATCH TYPE characteristic is set to PREFIX, 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. Table 8–9 shows examples of IP Domain addresses and associated route address matches, where the ADDRESS MATCH TYPE characteristic is set to PREFIX.

Table 8–9 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.

8.9.2.1 Overlapping IP Domains

RECEIVE DESTINATION Filters and Overlapping IP Domains

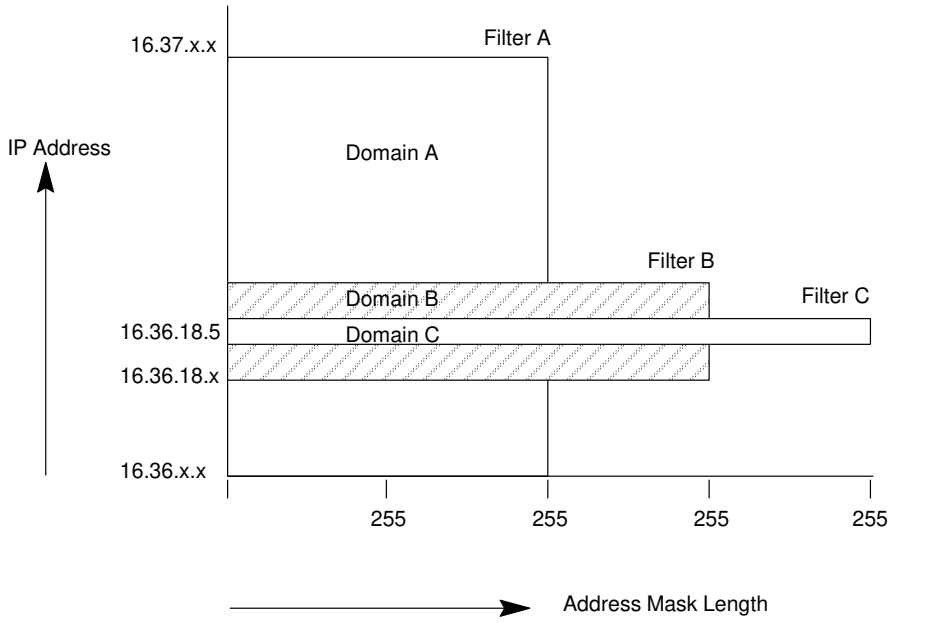
In Figure 8–10, Domain A has been set up for use with Filter A. Filter A has a filter action of BLOCK to ensure that routes 16.36.x.x are not accepted into the Routing Table. If you later decide that routes 16.36.18.x should pass, you could set up a domain (Domain B) for use with a filter (Filter B) 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 16.36.18.10 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 filters, 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 8–10 IP Domains and Filters



9
DECnet/OSI Routing

9.1 Introduction

This chapter describes how to manage DECnet/OSI routing on the DECNIS. It includes information on running the DECnet Phase IV routing protocols.

9.1.1 NCL Commands

All tasks performed using NCL commands assume that you have already logged on to a suitable host system on the network and started NCL, and have set default to the DECNIS as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

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.

9.2 DECNIS Addressing

9.2.1 Introduction

Network addresses in a DECnet/OSI network are called NSAP (Network Service Access Point) addresses. The *DECnet/OSI Routing Overview* describes NSAP addresses in detail.

The DECNIS can have from one to three NSAP addresses.

9.2.2 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.

9.2.3 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.

9.3 Deriving the Phase IV-Compatible NSAP Address

9.3.1 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.

9.3.2 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.

9.4 Changing the Phase IV Prefix

9.4.1 Restrictions

Reboot the DECNIS

To change the Phase IV prefix, you must run the configurator and then reboot the DECNIS; you cannot change the Phase IV prefix of a running system.

Use the Same Phase IV Prefix Throughout the Network

You must give all the DECNIS systems in your network the same Phase IV prefix.

9.4.2 Procedure

To change the Phase IV prefix of the DECNIS, supply information to the configurator as follows, and then reboot the DECNIS:

	Screen	Action
1	Sections Menu	Select Routing.
2	Routing Options Menu	Select Modify Routing Information.

	Screen	Action
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.

9.5 Changing the Phase IV Address

9.5.1 Reboot the DECNIS

To change the Phase IV address, you must run the load host configurator and the DECNIS configurator, and then reboot the DECNIS; you cannot change the Phase IV prefix of a running system.

9.5.2 Procedure

Change the Phase IV address of the DECNIS as follows:

1. Run the load host configurator to enter the new Phase IV address.
2. Run the DECNIS configurator to generate a new master NCL script file containing the new Phase IV address.
3. Reboot the DECNIS.

9.6 Deriving the Phase V-Style NSAP Address

9.6.1 Introduction

The DECNIS constructs its Phase V-style NSAP addresses from information you supply during configuration.

9.6.2 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.
- *System_ID* is the base hardware address for the DECNIS.

9.7 Changing the Phase V Area Address

9.7.1 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.

9.7.2 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}
   ```

9.8 NSAP Formats

9.8.1 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).

9.8.2 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.

9.8.3 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.

9.9 Deciding Which Routing Algorithms to Run

9.9.1 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.

9.9.2 Procedure

The *DECnet/OSI Routing Overview* describes how to choose the routing algorithm that the DECNIS should use. Table 9–1 summarizes the information provided there.

Table 9–1 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 9.16).

9.9.3 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.

9.10 Changing the Routing Algorithms

9.10.1 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 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.

9.10.2 Reboot the DECNIS

To change the routing algorithm that the DECNIS runs, either at Level 1 or Level 2, you must run the DECNIS configurator, and then reboot the DECNIS; you cannot change the routing algorithm of a running system.

9.10.3 Procedure

Change the routing algorithm(s) that the DECNIS runs by supplying information to the configurator as follows, and then rebooting 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.

9.11 Changing the Circuit Cost

9.11.1 Introduction

Each routing circuit has a Level 1 cost and, if applicable, a Level 2 cost associated with it (Section 6.22 describes routing levels). These costs are used by the routing algorithms in deciding which route through the network a packet should take.

9.11.2 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.

9.11.3 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.

9.12 Circuit Priority and Designated Routers

9.12.1 Designated Router Definition

The designated router for a LAN is the router whose routing circuit priority is highest.

9.12.2 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.

9.12.3 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.

9.12.4 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.

9.13 Changing the Circuit Priority

9.13.1 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.

9.13.2 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.

9.14 Manual Routing

9.14.1 Definition

Manual routing is used on routing circuits over which adaptive routing information is not exchanged. You must therefore provide this routing information, and change it manually if the network changes.

9.14.2 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 9.15.
- Circuits on routers running the link state routing algorithm connecting at Level 2 to routers running the routing vector algorithm: see Section 9.16.
- CSMA/CD routing circuits connecting to end systems that do not support automatic configuration as defined by ISO 9542: see Section 9.17.

9.15 Connecting to Foreign Routing Domains

9.15.1 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.

9.15.2 Configuration Restriction

Only Level 2 routers can connect to foreign routing domains.

9.15.3 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.

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 9.8).

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
```

9.16 Connecting to Routers Running a Different Routing Algorithm

9.16.1 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.

9.16.2 Definition: Interphase Link

An interphase link is an example of 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.

9.16.3 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.

9.16.4 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.

9.16.5 Restriction: Phase IV Area Numbers

The two areas connected by an interphase link must have different Phase IV area numbers.

9.16.6 Example

In the configuration shown in Figure 9–1, 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 to communicate between the link state domain (Area 1 and Area 2) and the routing vector domain (Area 9 and Area 10), you need to set up an interphase link between the DECNIS and the DECnet Phase IV router.

9.16.7 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.

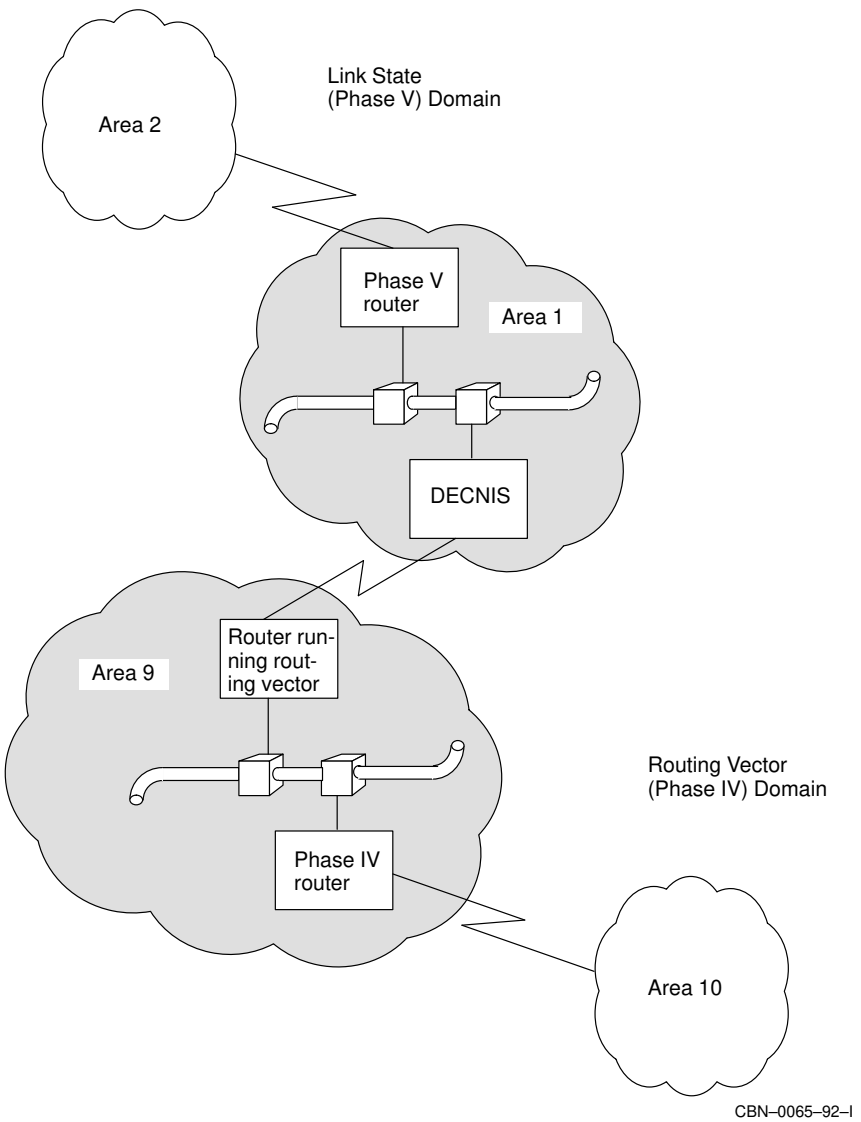
   - Specify the ADDRESS PREFIX for the reachable address; this is the Phase IV compatible area address as described in Section 9.3. See also Section 9.16.8.
   - Specify the cost of the reachable address: this must be greater than or equal to the circuit cost.
   - Specify the data format as PHASEIV, unless the receiving router is a DECNIS or another DECnet/OSI router (running the routing vector algorithm).

   Example

   In Figure 9–1, assume the name of the circuit is Salt, and the Phase IV prefix for the network is 49. Assume also that 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
```

Figure 9–1 Interphase Links



2. Create an inbound reachable address for each area in the link state domain.

Example

In Figure 9–1, 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
```

9.16.8 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 (aasuming a Phase IV prefix of 49::) are:

49::00-0
49::00-1
49::00-2
49::00-3

9.17 Connecting to LAN Systems Without Automatic Configuration

9.17.1 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.

9.17.2 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 9.2.
LAN-address	is the LAN hardware address for the end system.

9.17.2.1 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.

9.17.3 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
```

10
AppleTalk and IPX Routing

This chapter describes how to use the DECNIS for AppleTalk and NetWare IPX routing.

10.1 AppleTalk Routing

10.1.1 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.

10.1.2 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.

10.1.3 License Required

You need a PC LAN license to set up the DECNIS as an AppleTalk router.

10.1.4 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 10.3 explains tunnel circuits.

10.1.5 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.

10.1.6 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.

10.1.6.1 AppleTalk Addresses and Tunnel Circuits

Do not associate an AppleTalk address with a tunnel circuit.

10.1.7 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).

10.1.8 Setting Up AppleTalk Routing on a LAN Routing Circuit

10.1.8.1 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.

10.1.8.2 Procedure

Configure a LAN routing circuit to use AppleTalk routing as follows:

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 10.1.6 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
```

10.1.8.3 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
```

10.1.9 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.

10.1.9.1 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
```

10.1.10 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.

10.1.10.1 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.

10.1.10.2 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.

10.1.11 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.

You can use character strings for zone names that contain AppleTalk specific characters. Refer to your AppleTalk system documentation for full details of the AppleTalk characters. Select the character you want to use from the AppleTalk information and 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.

10.2 NetWare Routing

10.2.1 Introduction

The DECNIS supports the routing of NetWare IPX packets on CSMA/CD or FDDI circuits to and from the following:

- NetWare IPX routers
- NetWare IPX workstations (clients)
- NetWare IPX servers (file and print servers)

10.2.2 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.

10.2.3 License Required

You need a PC LAN license to set up the DECNIS as a NetWare IPX router.

10.2.4 Native and Tunnel Routing Circuits

There are two ways of using the DECNIS as a NetWare IPX router:

1. By using native IPX routing on CSMA/CD or FDDI circuits. These circuits route IPX packets to and from IPX end nodes.
2. By using tunnel circuits which encapsulate IPX packets in Internet Protocol (IP) packets. Tunnel circuits can use synchronous lines, as well as CSMA/CD or FDDI circuits. Section 10.3 explains tunnel circuits.

10.2.5 NetWare Addresses

A NetWare network number identifies a NetWare LAN. Each NetWare LAN must have a unique network number.

This number is assigned by the Novell administrator by selecting an unused hexadecimal number between 1 and FFFFFFFF with a maximum length of 8 digits.

The number identifies the LAN that circuits connect to. Several circuits can use the same address if they are all using the same data link encapsulation (framing) on all the circuits that are connected to the LAN. Typically, each NetWare IPX circuit connects to a different LAN, and therefore has a different network number.

Circuits connected to the same LAN with different data link encapsulation must have different (unique) network addresses. Different circuit types must use different network numbers, so an FDDI circuit has a different network number than a CSMA/CD circuit.

NetWare workstations learn their network number from NetWare servers or NetWare routers.

10.2.6 Setting Up NetWare IPX Routing on a LAN Routing Circuit

10.2.6.1 FDDI Restriction

On FDDI circuits, the NETWORK PROTOCOL of type NETWARE IPX 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.

10.2.6.2 Procedure

Configure a LAN routing circuit to use NetWare IPX routing as follows:

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, irrespective of the number of routing circuits on the DECNIS that will use IPX routing.

2. Create a NETWORK PROTOCOL of type NETWARE IPX for the LAN routing circuit on which you want to use 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 10.2.5 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
   ```

10.2.6.3 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 -
_NCL> NETWARE IPX

NCL> DISABLE ROUTING CIRCUIT circuit-name
```

10.2.7 Frame Types on LAN Circuits

You must specify the frame type to be used by IPX on the LAN circuit.

Table 10–1 describes what frame type to use when using NetWare circuits on the DECNIS.

Table 10–1 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.

10.2.7.1 Changing the Frame Type on a Routing Circuit

Set the type of frame for IPX routing on a routing circuit as follows:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX

NCL> SET ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX -
_NCL> NETWARE DATA LINK ENCAPSULATION type

NCL> ENABLE ROUTING CIRCUIT circuit-name -
_NCL> NETWORK PROTOCOL NETWARE IPX
```

10.2.8 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 and 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 the interval.

10.2.8.1 Changing the Periodic Routing Protocol Timer

Change the value of this timer by issuing the following commands:

```
NCL> DISABLE ROUTING CIRCUIT circuit-name -
_NCL> 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-
_NCL> PROTOCOL NETWARE IPX
```

10.2.9 NetBIOS Broadcast

NetBIOS™ is a separate 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.

10.2.9.1 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
```

mode is ON or OFF.

10.3 Tunnel Circuits

10.3.1 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 its 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.

10.3.2 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.

10.3.3 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.

10.3.4 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. You supply the system IP address either during load-host configuration (for BOOTP load hosts) or in the Routing section of the configurator (for MOP-only load hosts).

10.3.5 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
```

10.3.6 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 11 describes how to set up and manage DTEs, and use X.25 gateway functions of the DECNIS.
- Chapter 12 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 13 explains how to set up incoming and outgoing X.25 security on the DECNIS.

11

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 already logged on to a suitable host system on the network and started NCL, and have set default to the DECNIS as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

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.

11.1 Introduction

11.1.1 General

When using X.25 gateway 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 13)

11.1.2 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.

11.1.3 Incoming Calls

Each DECNIS has one or more filters. Each filter is associated with a Client system by means of a server client.

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.

11.2 Managing DTEs

11.2.1 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.

11.2.2 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 11.2.4.
8. Set the holdback timer for the link: see Section 11.2.5.
9. Set the window size for the link: see Section 11.2.6.
10. Set the maximum frame size for the link: see Section 11.2.7.

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 1.11.

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 11.3). Note that, if it does not already exist, you must create a DTE class with this name: see Section 11.3.2.

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 11.6).

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 11.2.10.1.

21. Add the DTE to a DTE class on your system: see Section 11.3.4. This will usually be the DTE class specified in step 17.

22. Enable the DTE:

    ```
    NCL> ENABLE X25 PROTOCOL DTE DTE-name
    ```

11.2.3 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. *DEC Network Integration Server Installation and Configuration* explains how to prevent lines associated with a network interface card from attempting to load or dump.

11.2.4 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.

11.2.5 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 11.2.4.

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 that specifies the time in milliseconds.

Note that when you change the value of this timer on a running system, the timer is reset.

11.2.6 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.

11.2.7 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.

11.2.8 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.

11.2.8.1 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.

11.2.9 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 11.7.4.
4. Amend the list of members of the DTE class of which this DTE was a member: see Section 11.3.4.
5. Delete the DTE class, if it is now empty: see Section 11.3.3.
6. Disable and delete the LAPB link associated with this DTE:

   ```
   NCL> DISABLE LAPB LINK link-name
   NCL> DELETE LAPB LINK link-name
   ```

11.2.10 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.

11.2.10.1 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
   ```

11.2.10.2 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 charactreistic to TRUE if the profile you specified for the DTE supports extend packet sequence numbering, for example ISO8208_E.

11.3 Managing DTE Classes

11.3.1 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.

11.3.2 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 11.3.4.
4. Set up security for this DTE class: see Chapter 13.

11.3.3 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 11.2.9), or put them in another DTE class (see Section 11.3.4).
2. Ensure that no enabled DTEs specify the DTE class as their INBOUND DTE CLASS characteristic (see Section 11.2.2).
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
   ```

11.3.4 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 11.2.2.

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
   ```

11.4 Managing Server Clients

11.4.1 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 11.5).

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 11–1 illustrates the operation of filters and server clients on the DECNIS.

11.4.2 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 11.4.4.

5. Create the filters specified above, if necessary: see Section 11.5.2.

6. Enable the CLIENT entity:

   ```
   NCL> ENABLE X25 SERVER CLIENT client-name
   ```

11.4.3 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 11.5.3.

11.4.4 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 1.9).

11.4.5 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
   ```

11.5 Managing Filters

11.5.1 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 6.14. Figure 11–1 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 11–1, 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.

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 11–1, 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.

11.5.2 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 11.5.4.
4. Set up security for the filter: see Chapter 13.

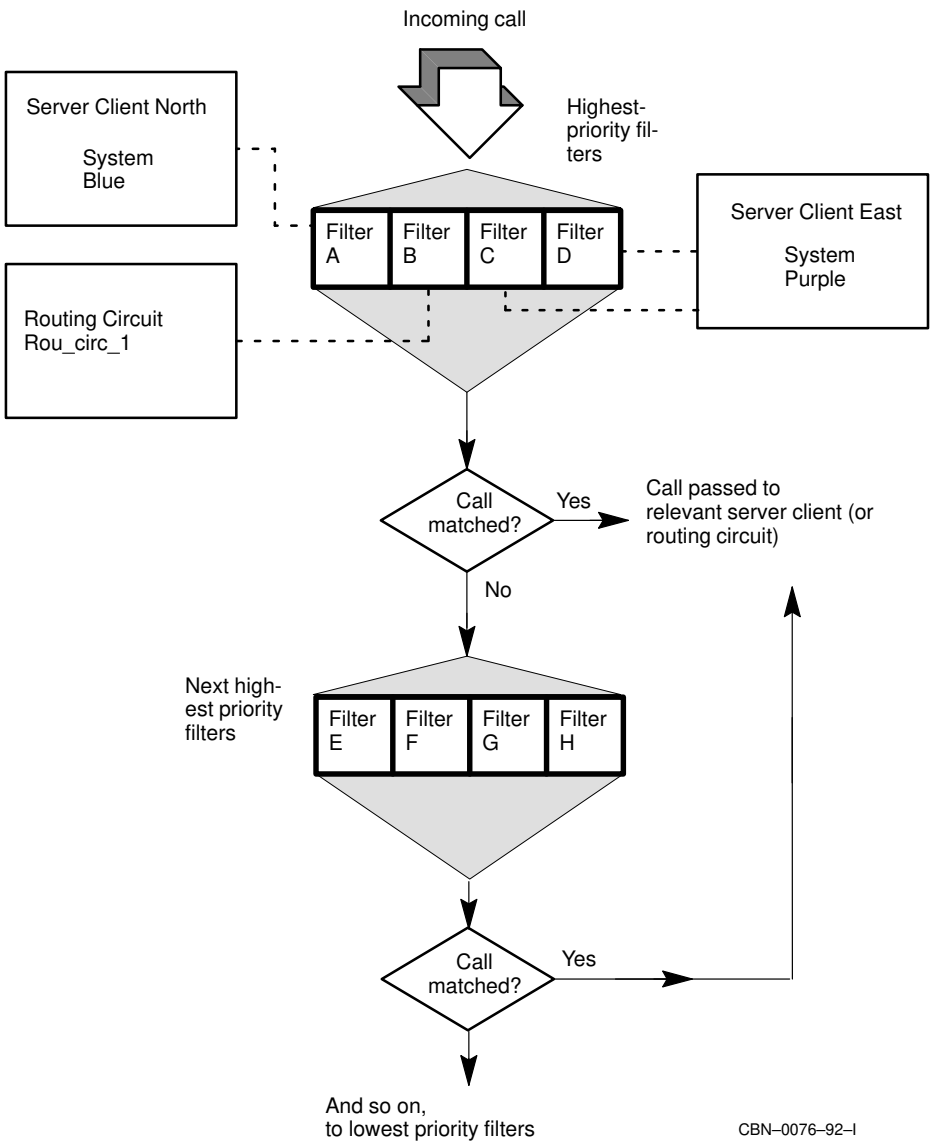
11.5.3 Deleting a Filter

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 11.4.5.
2. Delete the FILTER entity:

   ```
   NCL> DELETE X25 ACCESS FILTER filter-name
   ```

Figure 11–1 How Filters Handle Incoming Calls on the DECNIS



11.5.4 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".

11.6 Managing PVCs

11.6.1 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.

11.6.2 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 11.2.2.
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 13.

11.6.3 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
```

11.7 Managing Groups

11.7.1 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.

11.7.2 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 11.7.4.

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.

11.7.3 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
   ```

11.7.4 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.

12
X.25 Relay

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.

All tasks performed using NCL commands assume that you have already logged on to a suitable host system on the network and started NCL, and have set default to the DECNIS as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

All tasks performed using the configurator assume that you have started the configurator on a suitable host system on the network, selected Modify from the Main Menu, and selected the DECNIS you wish to modify from the DECNIS Nodes screen.

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.

12.1 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 12.7).

Figure 12–1 Principles of X.25 Relay

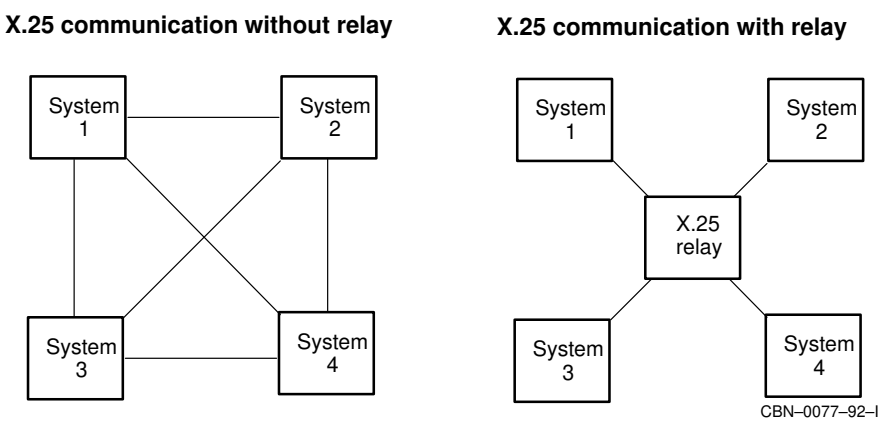
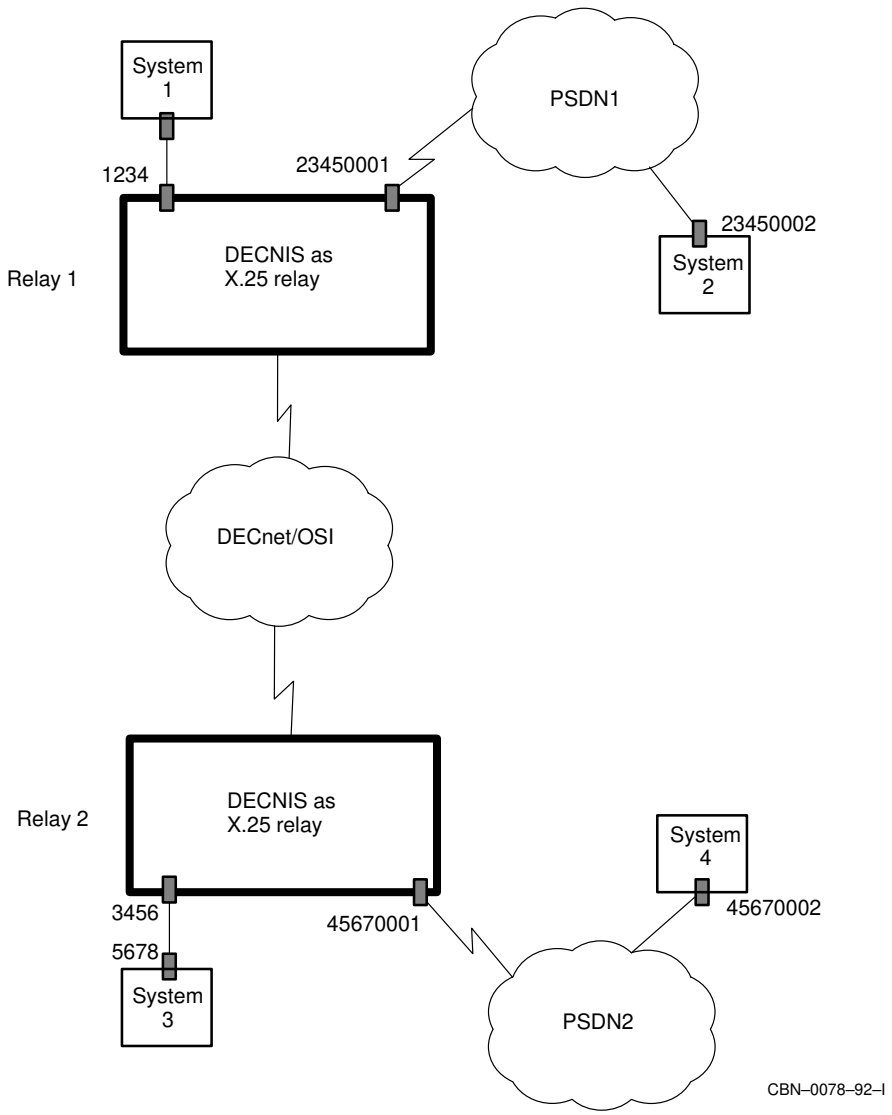


Figure 12–1 illustrates the principle of X.25 relay operation. Systems 1, 2, 3, and 4 wish to communicate using X.25 software. In the left diagram, each system sets up a point-to-point X.25 link with every other system. In the right diagram, the same connectivity is achieved, using fewer lines and DTEs, 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 12–2 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 12–2 DECNIS Performing More Complex Relay Functions



12.2 How the DECNIS Operates as an X.25 Relay

12.2.1 Introduction

An X.25 relay connects to X.25 networks using DTEs; Section 11.2.2 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.

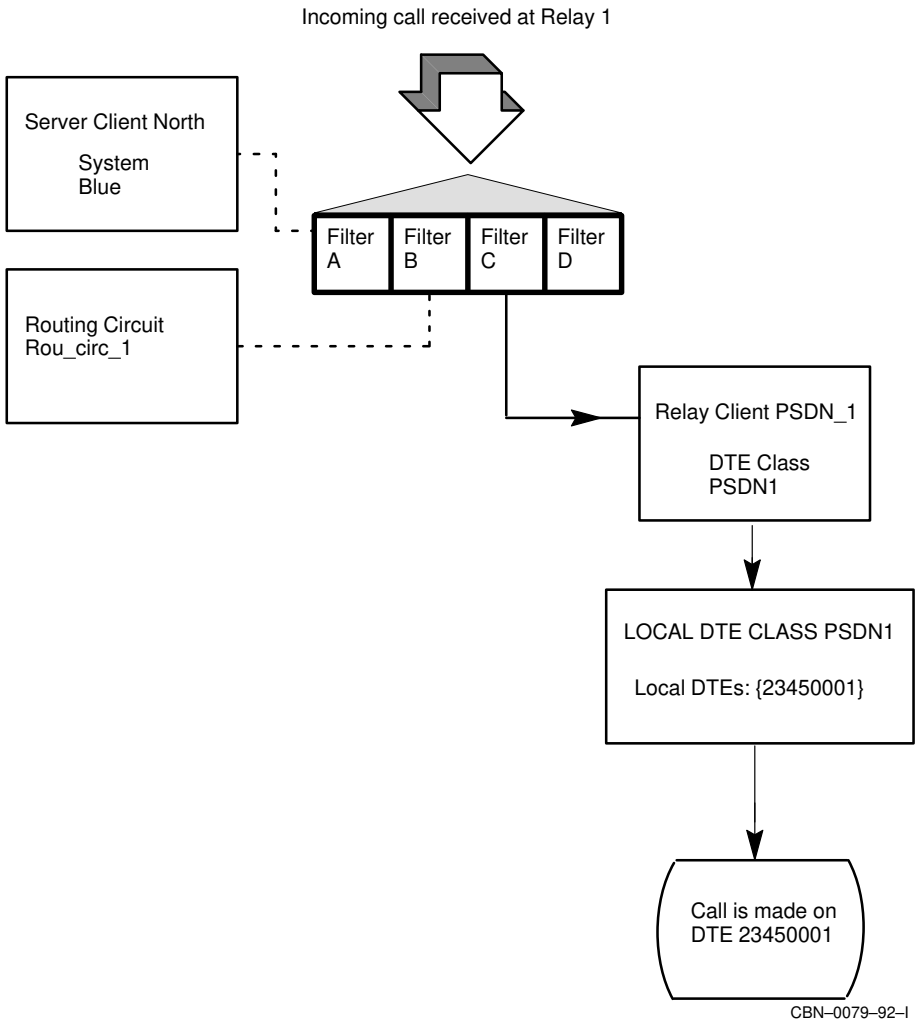
For example, for System 1 to make an X.25 call to System 2, the following occurs (see Figure 12–3):

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 11.1.3).
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 11.1.3).
3. This filter is associated with the relay client that Relay 1 is using to switch calls to System 1.

Figure 12–3 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.

12.2.2 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.

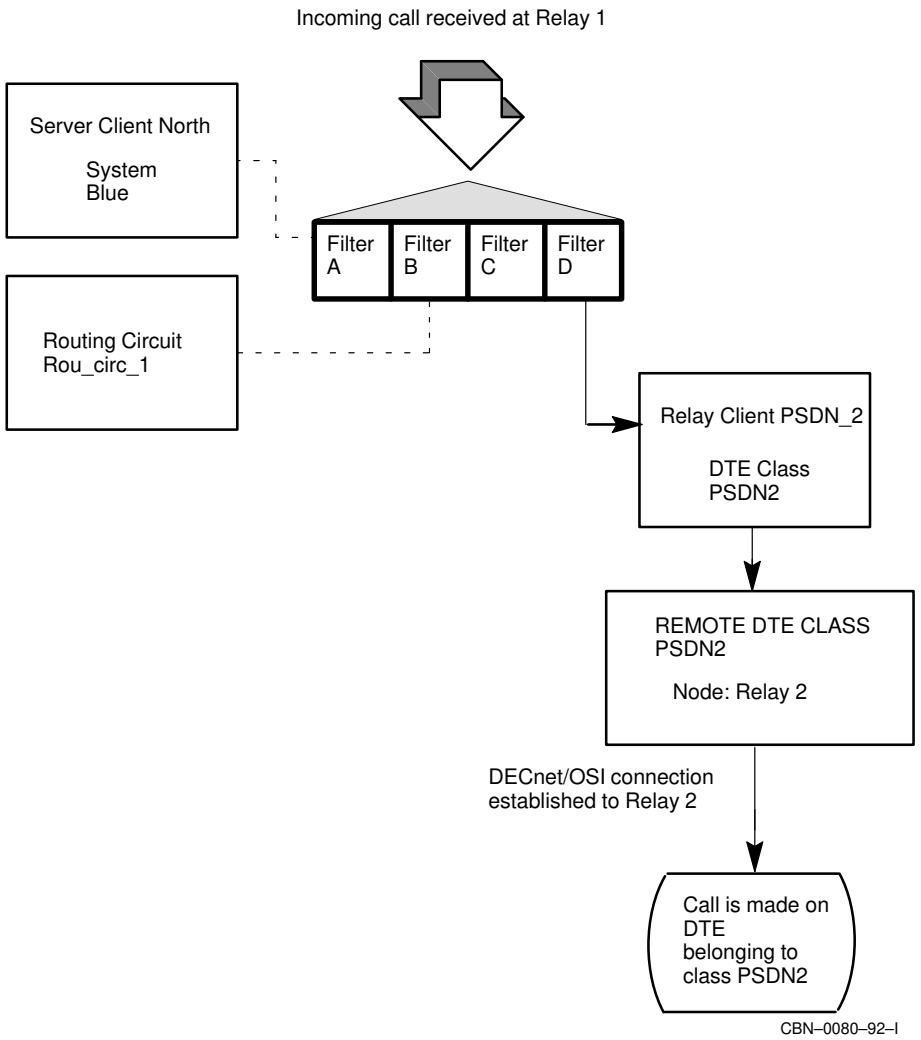
For example, Figure 12–4 shows the mechanism for System 1 to communicate with System 4:

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 11.1.3).
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.

12.2.3 Using X.25 Reachable Addresses to Translate NSAP Addresses

If, in Figure 12–2, 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 12.9 describes X.25 reachable addresses.

Figure 12–4 DECNIS Switching Calls Remotely



12.3 Managing Relay Clients

You must set up a relay client for every DTE or PSDN to which X.25 calls are to be switched.

12.3.1 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 11.5 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 11.3 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 12.4 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 6.12 describes templates and how to manage them.

6. Set up security for the relay client; see Section 13.5.

7. Create an X.25 reachable address for the relay client if required, to translate target NSAP addresses to target DTE addresses; Section 12.9 describes X.25 reachable addresses and how to manage them.

8. Enable the relay client:

```
NCL> ENABLE X25 RELAY CLIENT client-name
```

12.3.2 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 11.5.3.

12.3.3 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
   ```

12.4 Managing Remote DTE Classes

12.4.1 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.

12.4.2 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 1.9), of the X.25 system that has a local DTE class called *class-name*.

12.4.3 Deleting a Remote DTE Class

Delete a remote DTE class in exactly the same way as deleting a local DTE class: see Section 11.3.3.

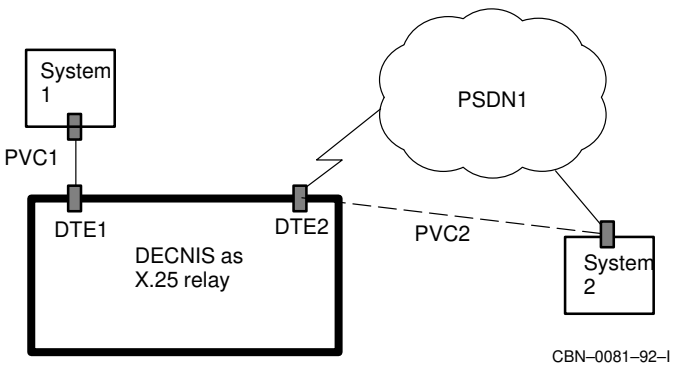
12.4.4 Changing the System Associated with a Remote DTE Class

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 1.9.

Figure 12–5 Switching Between PVCs Locally



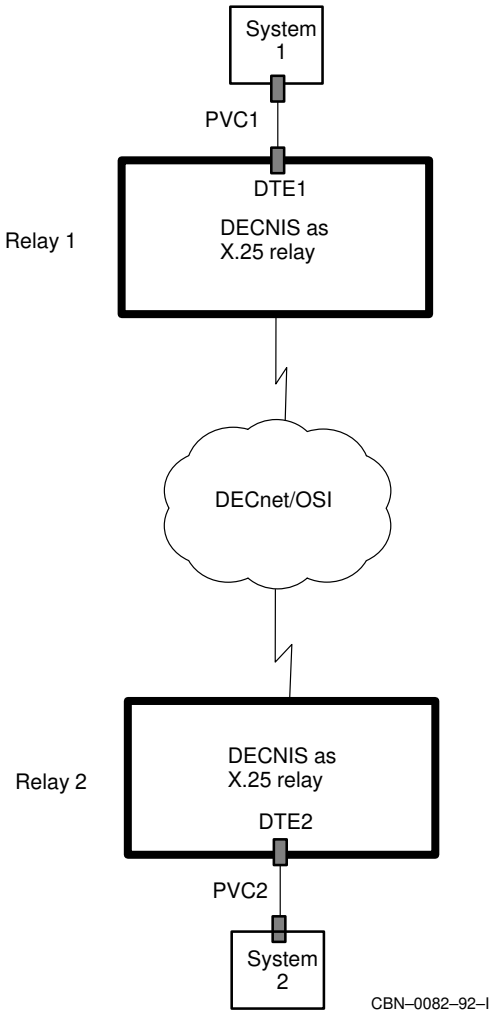
12.5 Creating a Permanent Association Between Two DTEs

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 11.6) on each of the X.25 connections, and setting up a relay PVC on the DECNIS.

Figure 12–5 and Figure 12–6 show two possible configurations. In Figure 12–5, 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 12–6, 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 12–6 Switching Between PVCs Remotely



12.6 Managing Relay PVCs

12.6.1 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 11.6 describes PVCs and how to manage them.

12.6.2 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 12–5, 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 12–6, this would be PVC2.

   ```
   NCL> SET X25 RELAY PVC relay-pvc-name REMOTE PVC remote-pvc
   ```

5. If the PVC called *remote-pvc* does not exist on your DECNIS (as in Figure 12–6), 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 PVC exists. Section 12.4 describes remote DTE classes, and how to manage them.

6. Set up security for the relay PVC; see Section 13.5.

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 12–6, you should create the RELAY PVC entity only on Relay 1 or Relay 2, not on both.

12.6.3 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
```

12.7 LLC2 Communication and the CONS LAN/WAN Relay

12.7.1 Introduction

Figure 12–7 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).

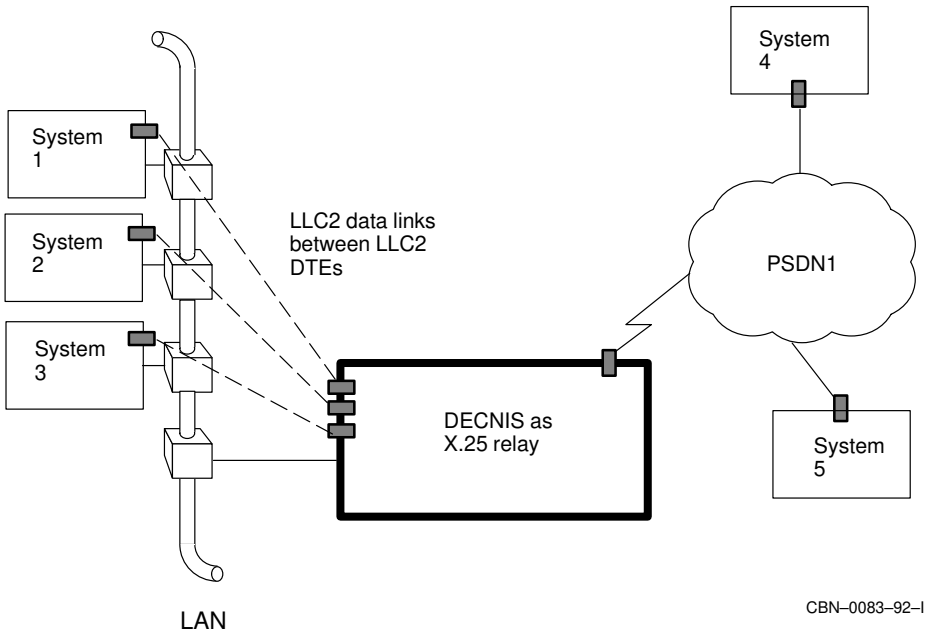
In addition, the DECNIS can also switch calls between the host systems on the LAN, for example, between System 1 and System 2.

12.7.2 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 12.8.3 describes how to set up LLC2 DTEs.

Figure 12–7 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 12–7, 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 12.3.1 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 12.9 describes X.25 reachable addresses and how to manage them.

12.8 Managing LLC2 DTEs

12.8.1 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 11.2.2). 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.

12.8.2 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 5.6 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 5.6).

   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 5.7).

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
```

12.8.3 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 11.2.2 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 11.3).

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 11.3. 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
    ```

12.8.4 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 11.3.4.

3. Delete the DTE class, if it is now empty: see Section 11.3.3.

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
```

12.8.5 Changing LLC2 DTE Characteristics

Change LLC2 characteristics in the same way as changing synchronous DTE characteristics: see Section 11.2.10.

12.9 Managing X.25 Reachable Addresses

12.9.1 Introduction

Create X.25 reachable addresses on the DECNIS to translate target NSAP addresses into target DTE addresses.

Consider the situation in Figure 12–7. 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.

12.9.2 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 6.13), 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 12.3.1.

12.9.3 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
```

13

X.25 Security

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.

All tasks performed using NCL commands assume that you have already logged on to a suitable host system on the network and started NCL, and have set default to the DECNIS as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

All tasks performed using the configurator assume that you have started the configurator on a suitable host system on the network, selected Modify from the Main Menu, and selected the DECNIS you wish to modify from the DECNIS Node screen.

13.1 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 13–1).
2. These rights identifiers are then checked against the ACL (Access Control List) associated with the filter that matches the call (see Figure 13–2). 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 13.6 describes ACLs.

Figure 13–1 How an Incoming Call Acquires Rights Identifiers

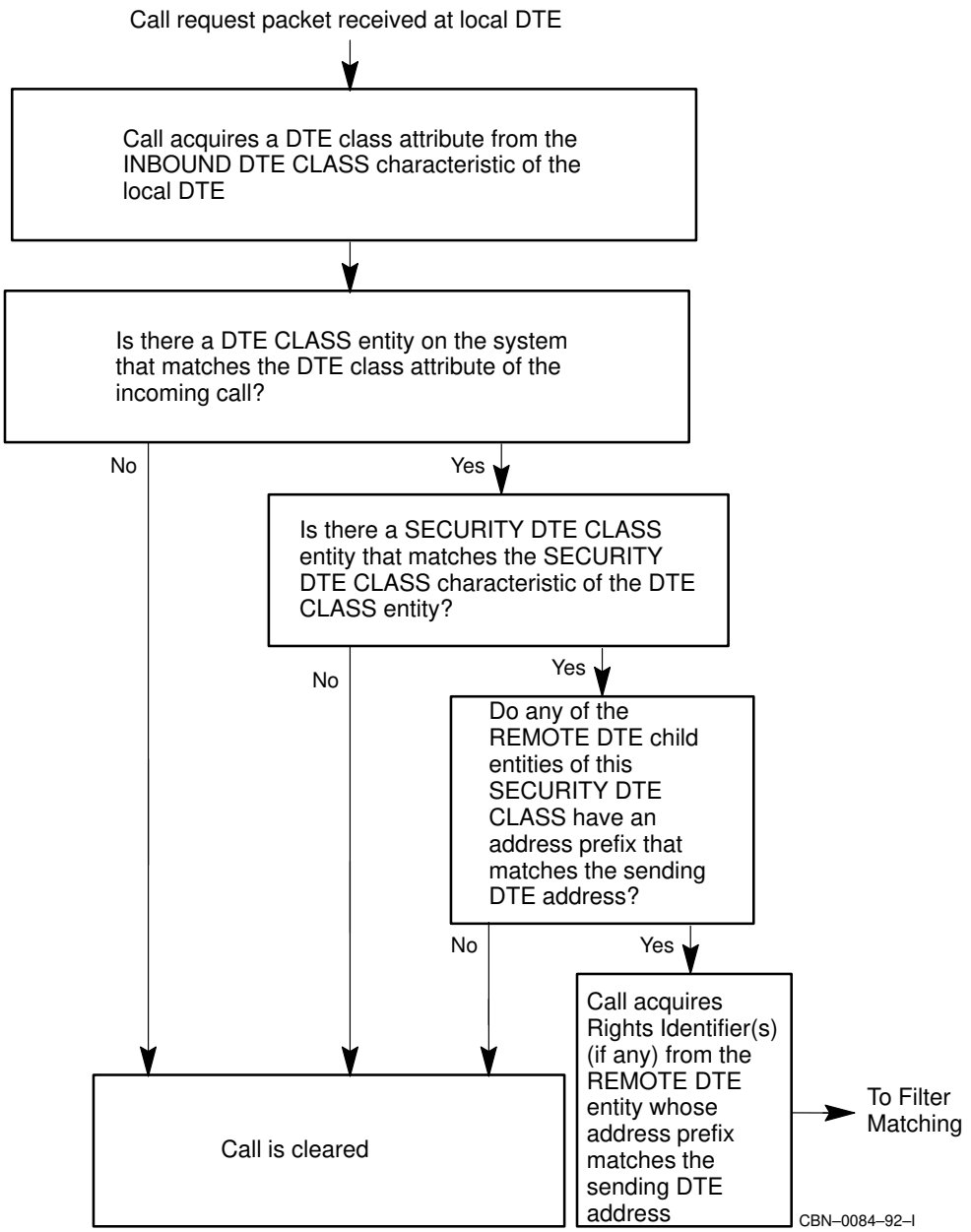
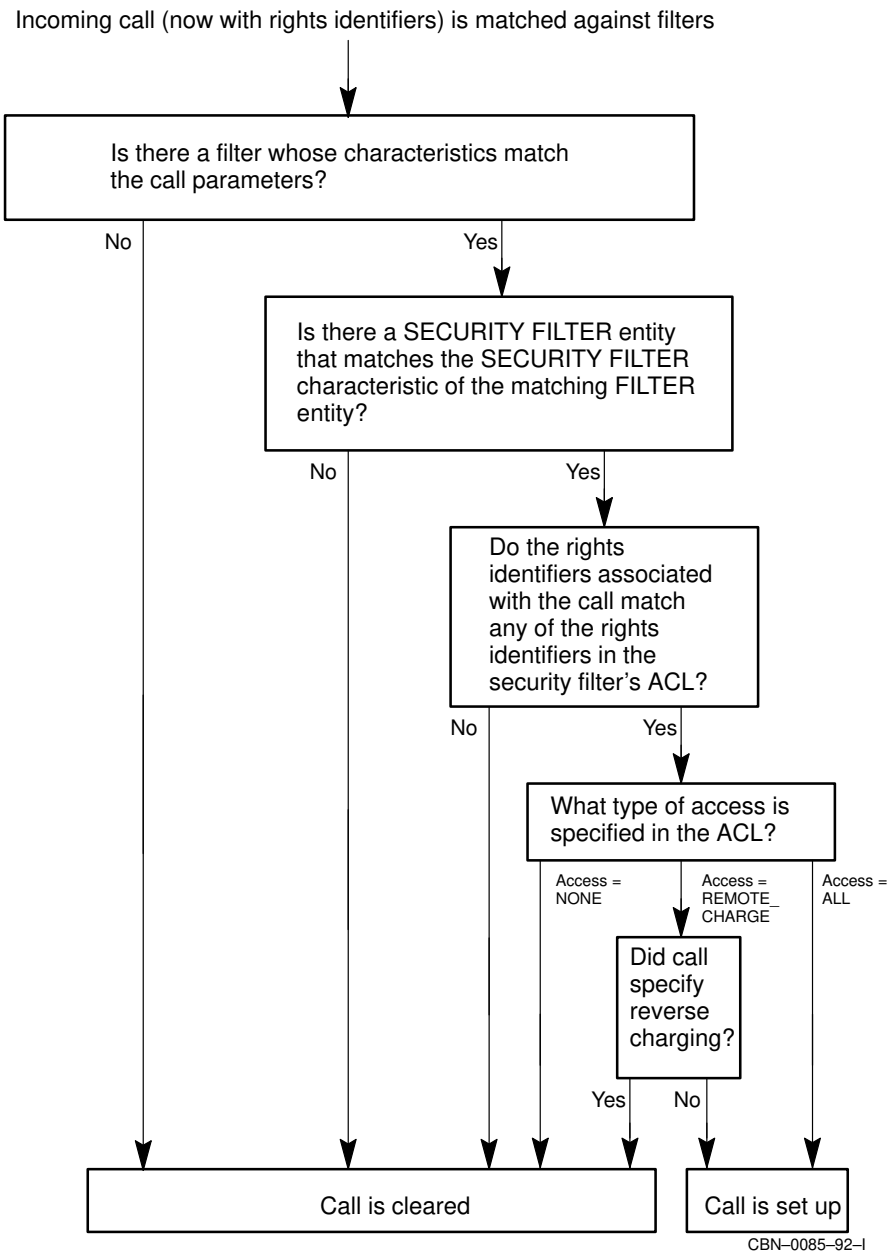


Figure 13–2 How X.25 Security Decides Whether to Permit an Incoming Call



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.

13.2 Setting Up Incoming Security

13.2.1 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.

13.2.2 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 on your system 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 a remote address prefix, then 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-name*, and a rights identifier for those that are to have remote-charge access: for example, *filter-name*-FREE and *filter-name*-REMOTE.

5. Now 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-name-FREE}

NCL> ADD X25 ACCESS SECURITY DTE CLASS security-DTE-class -
_NCL> REMOTE DTE remote-DTE-2 -
_NCL> RIGHTS IDENTIFIERS {filter-name-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-name -
_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-name-FREE), ACCESS = ALL), -
_NCL> (IDENTIFIER = (filter-name-REMOTE), ACCESS = REMOTE))
```

13.2.3 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.

13.2.4 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).

13.2.4.1 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 13.2.2. 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.

13.2.5 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 to the configurator 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 on the DECNIS.
4		Select another server client, to which this security information will be copied.

	Screen	Action
5	Incoming Security: X.25 Server Clients Options Menu	Select Copy Security Information From an X.25 Server Client again, and select another server client to which this security information will be copied. Continue with this process until all server clients have the same incoming security information.

13.2.6 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".

13.3 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.

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.

For outgoing security, each Client system that has the potential to use the DECNIS as a Connector system is granted rights identifiers. These rights identifiers are then checked against the ACL (see Section 13.6) of the remote DTE it wants to call. The ACL specifies the type of access that the Client system may 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 13–3 illustrates outgoing security for X.25 gateway functions on the DECNIS.

Figure 13–4 illustrates outgoing security on PVCs for X.25 gateway functions on the DECNIS.

Figure 13–3 How Outgoing Security Operates on the DECNIS

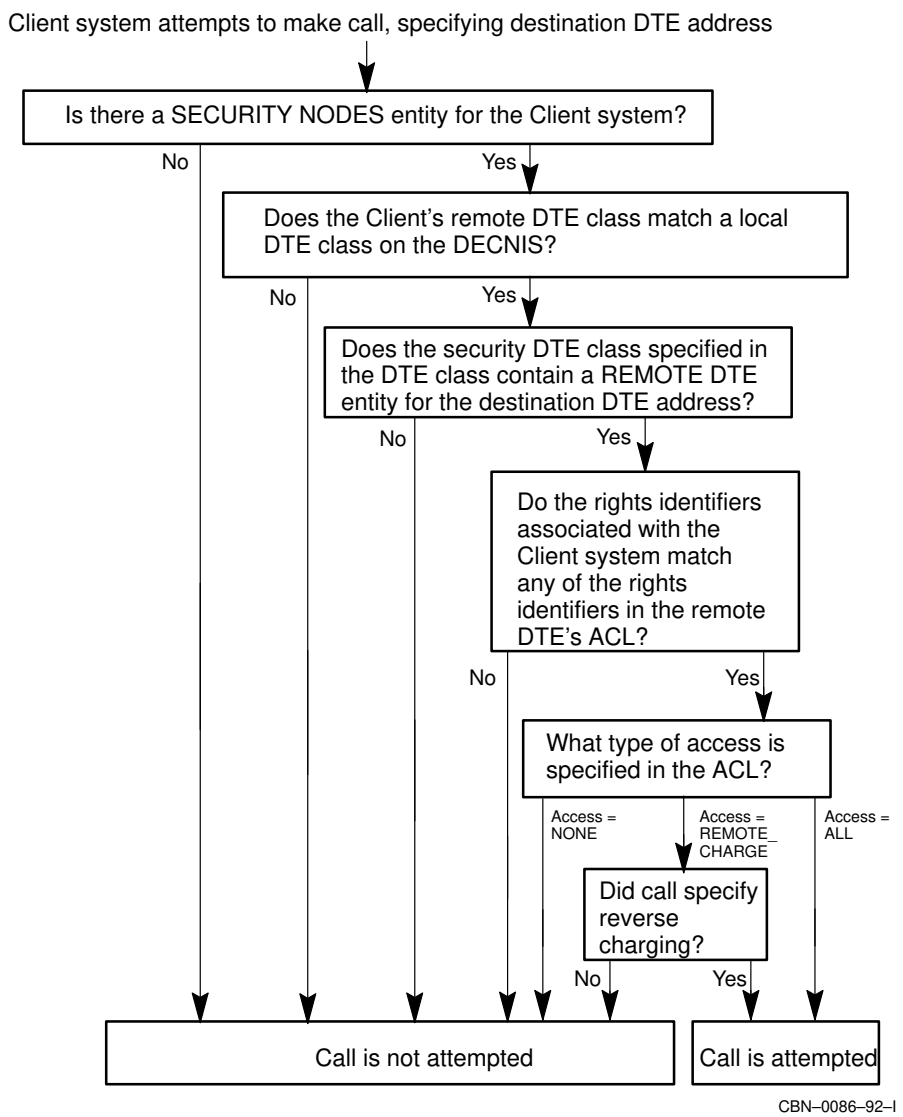
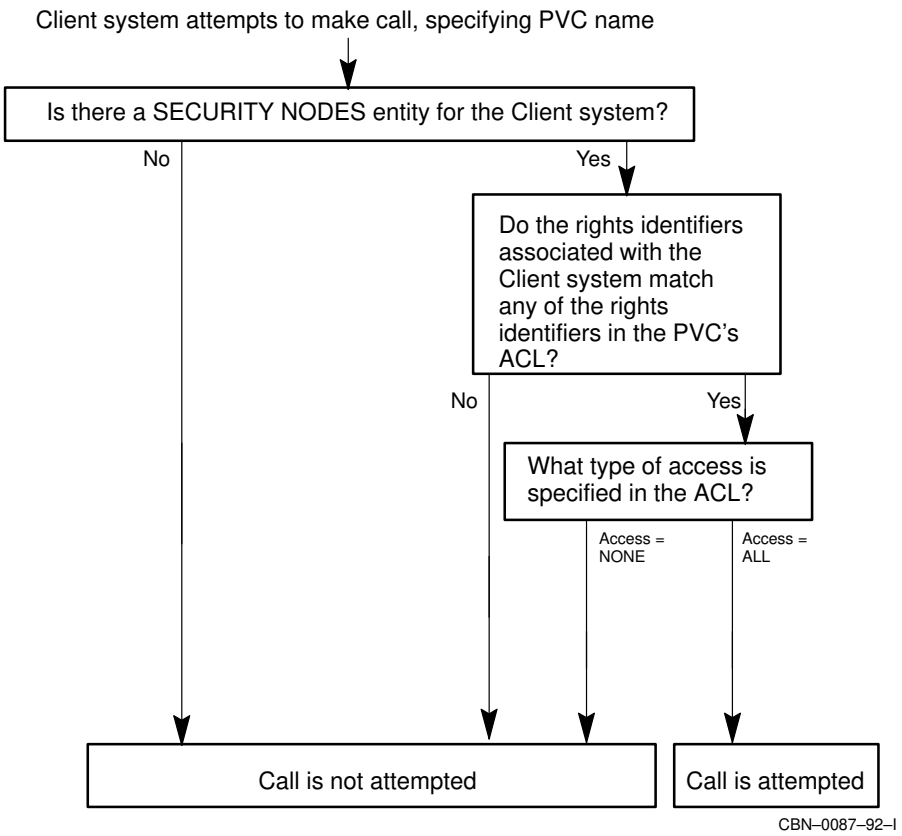


Figure 13–4 How Outgoing Security for PVCs Operates on the DECNIS



13.4 Setting Up Outgoing Security

13.4.1 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.

13.4.2 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 1.9).

   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))
```

13.4.3 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 13.4.2.
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.

13.4.4 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
```

13.4.5 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 1.9).

   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}
   ```

13.4.6 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).

13.4.6.1 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.

13.4.7 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))
```

13.4.8 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 configurator to set up outgoing security, all the DTE classes will use the same security DTE class, called "Default".

13.5 X.25 Relay Security

13.5.1 How X.25 Relay Security Works

Incoming security for X.25 relay functions works as described in Section 13.1. 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 13.2 describes how to do this.

Outgoing security for X.25 relay functions is almost identical to that described in Section 13.3. Set up X25 ACCESS SECURITY DTE CLASS REMOTE DTE entities, and associated ACLs, as described in Section 13.4. 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 REMOTE 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 13.4.2).

Figure 13–5 illustrates outgoing security for X.25 relay functions on the DECNIS.

For example, consider Figure 13–6. In this case, System 1 can make calls to System 3, but System 2 cannot. Set up security on the DECNIS as follows.

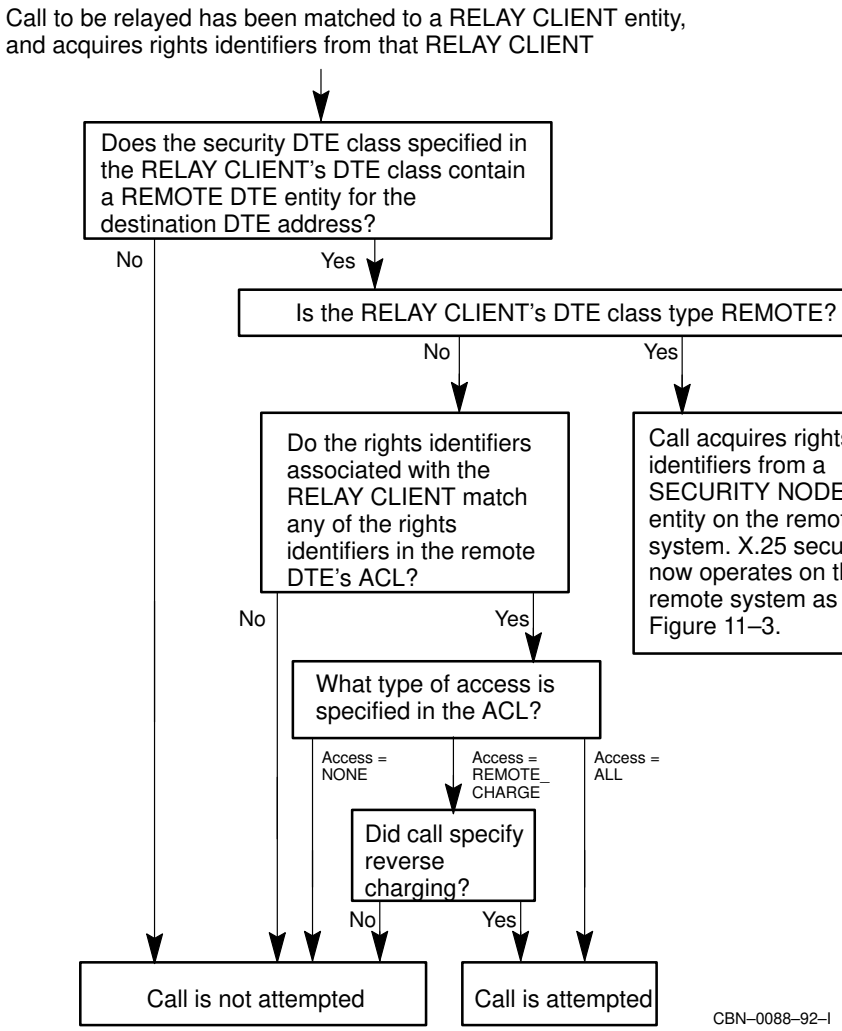
1. Set up incoming security to allow System 1, but not System 2, to access the filter(s) associated with the RELAY CLIENT entity corresponding to System 3 or PSDN1; see Section 13.2.2.
2. Grant the RELAY CLIENT entity corresponding to System 3 or PSDN1 a rights identifier; for example:

```
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)
```

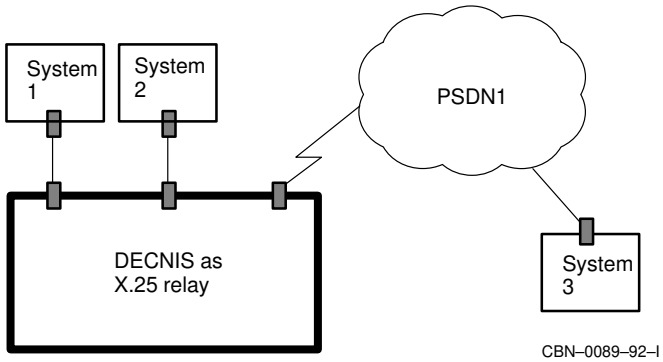
Figure 13–5 How Outgoing Security for X.25 Relay Functions Operates on the DECNIS



13.5.2 X.25 Relay Security and PVCs

Figure 13–7 illustrates outgoing security on PVCs for X.25 relay functions on the DECNIS.

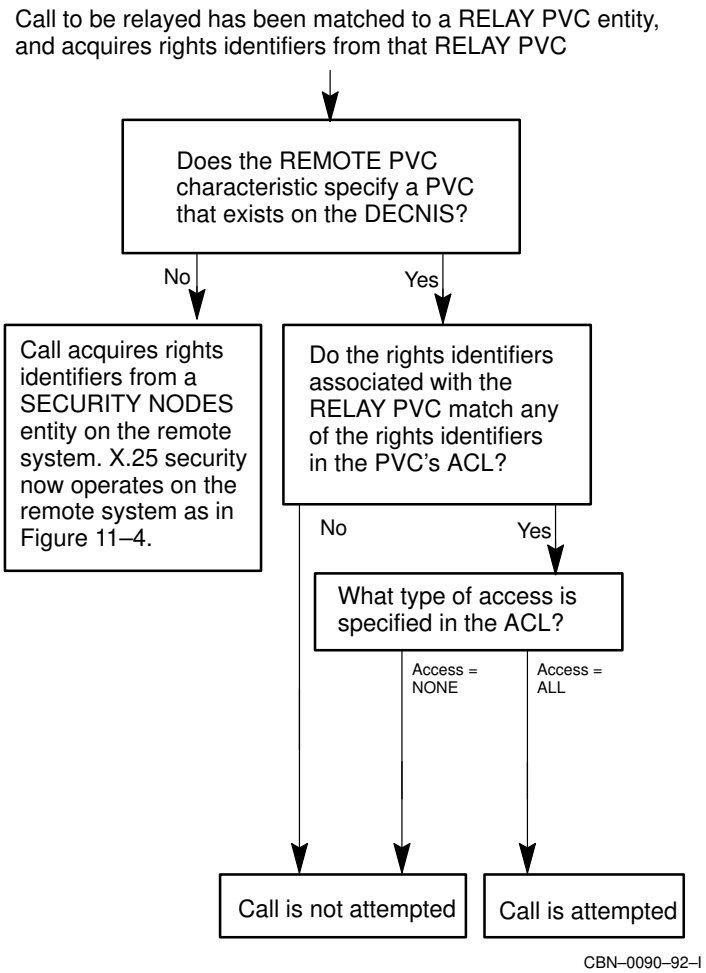
Figure 13–6 X.25 Security for Relay Functions



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 13–7 How Outgoing Security for X.25 Relay Functions on PVCs Operates on the DECNIS



13.6 ACLs

13.6.1 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.

13.6.2 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.

13.6.3 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 13.6.2 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 14 introduces the principles of bridging on the DECNIS, and describes how to set up bridge ports.
- Chapter 15 describes how to configure a bridge port to filter certain protocols and addresses.

14

Setting Up the DECNIS as a Bridge

14.1 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.

14.1.1 NCL Commands

All tasks performed using NCL commands assume that you have already logged on to a suitable host system on the network and started NCL, and have set default to the DECNIS as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

14.2 Bridging

14.2.1 Definition

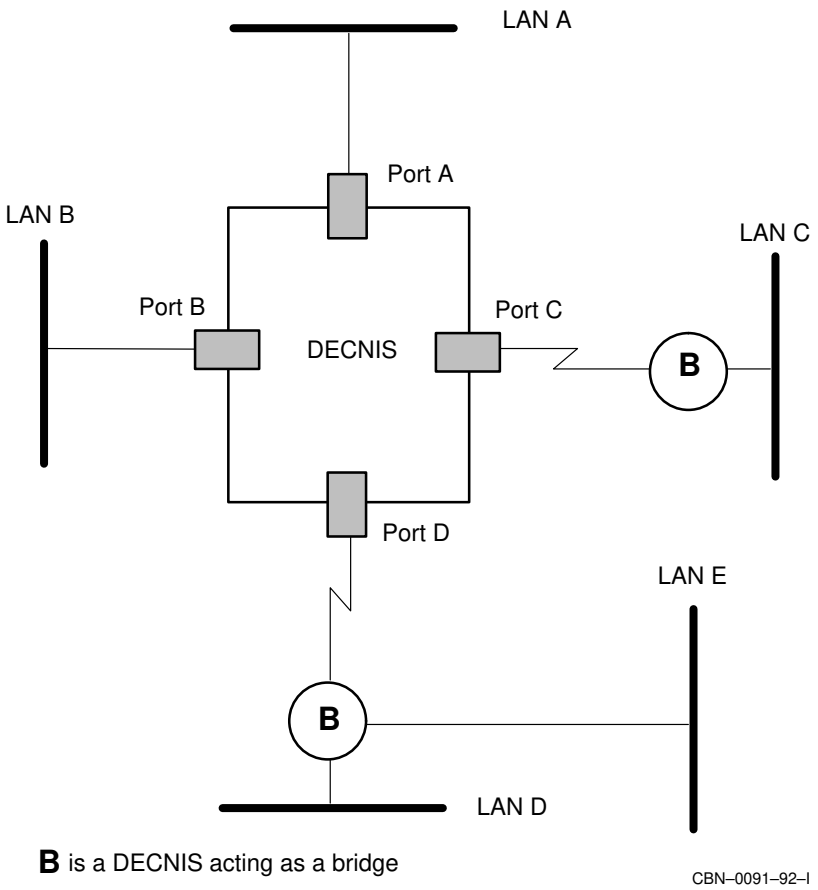
Bridging means linking two or more Local Area Networks (LANs) at the data link level to form an **extended LAN**.

A **bridge** is a server connected to two or more LANs that forwards frames according to source address, destination address, and protocol type.

14.2.2 Example

Figure 14–1 shows the DECNIS linking several LANs to form an extended LAN. The DECNIS is performing local and remote bridging on the extended LAN.

Figure 14–1 DECNIS Operating as a Bridge



14.2.3 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.

14.2.4 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.

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.

14.2.4.1 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 5.8 describes how to set up a VCP data link.

14.2.5 Extended LANs: Definition

LANs A, B, C, D, and E in Figure 14–1 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.

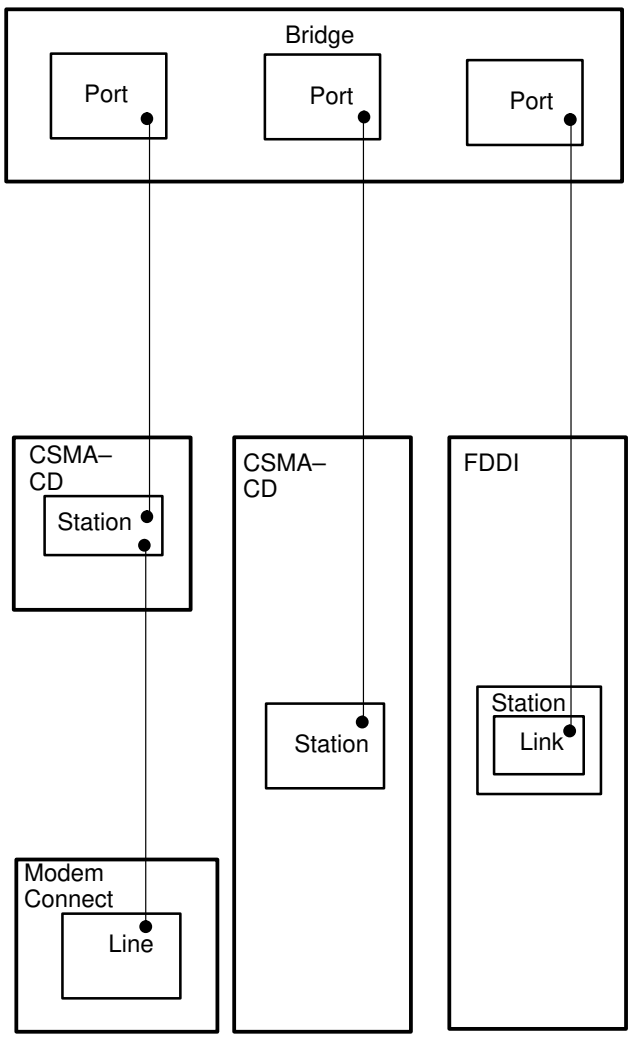
14.2.6 Further Reading

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.

14.3 Entities Associated with a Bridge Port

Figure 14–2 and Figure 14–3 show the entities required by each type of bridge port.

Figure 14–2 Entities Required by Local and Remote VCP Bridge Ports

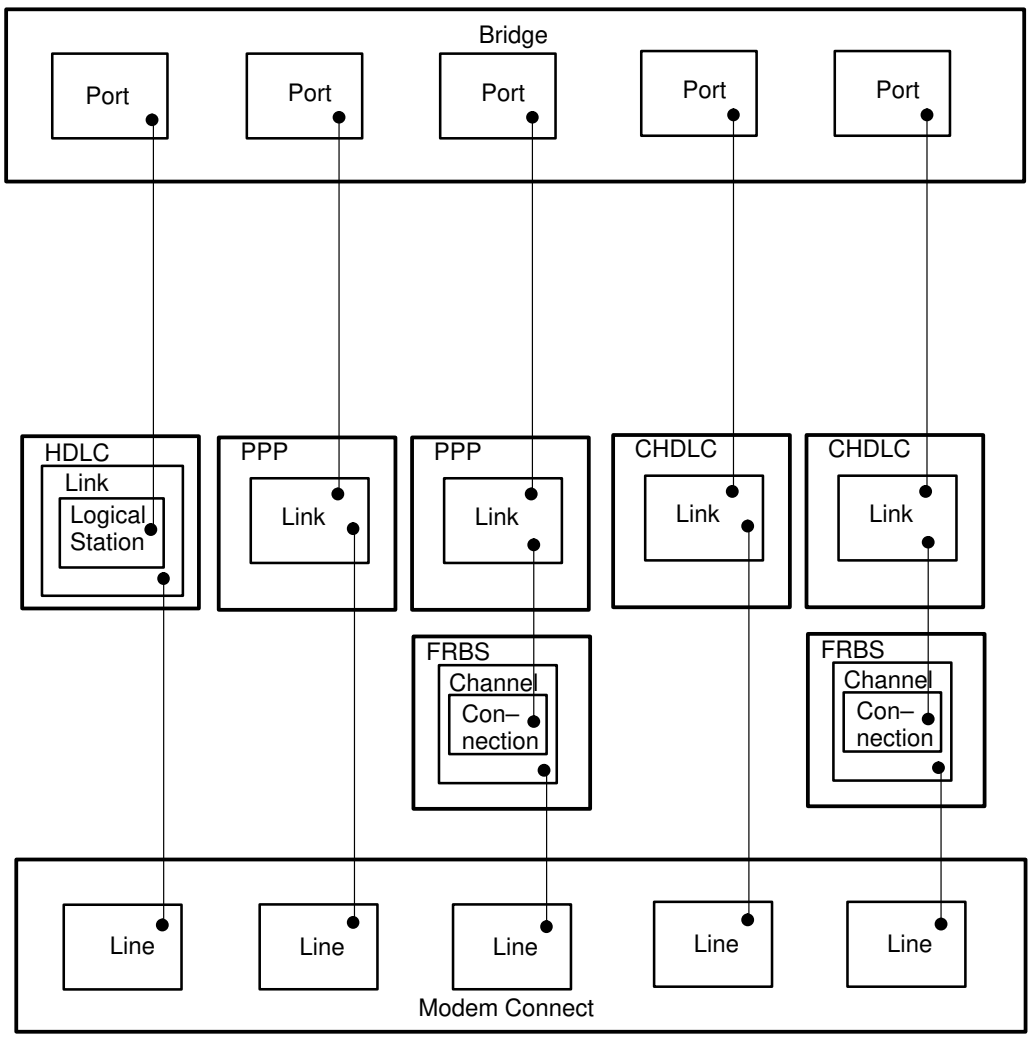


An association between two entities (indicated by a line connecting them) is made by specifying the name of the lower entity as a characteristic value of the upper entity.

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 14–3 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

14.4 Routing and Bridging of Protocols

14.4.1 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

14.4.2 Protocols that Can Be Bridged or Routed

The DECNIS will bridge or route the following protocols, depending on its configuration:

- IP
- AppleTalk
- NetWare IPX

14.4.3 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.

14.4.4 Switching Between Routing and Bridging

To alter how the DECNIS treats these protocols, run the DECNIS 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).

14.4.5 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.

14.4.6 Protocol Filtering

If these protocols are not routed, you can still prevent them from being bridged by setting up bridge filtering: see Chapter 15.

14.5 Setting Up the DECNIS as a Bridge

14.5.1 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.

14.5.2 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 an HDLC, PPP, CHDLC, or VCP connection that uses a DEC WANcontroller 622 Network Interface Card.

- Section 5.9 describes how to set up an HDLC connection.
- Section 5.17 describes how to set up a PPP connection.
- Section 5.18 describes how to set up a CHDLC connection.
- Section 5.8 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.

14.5.3 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 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.

14.6 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.

14.7 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.

14.8 Deleting a Bridge Port

14.8.1 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.

14.8.2 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 15.14):

   - 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 15.15).

3. Delete the bridge port:

   ```
   NCL> DELETE BRIDGE PORT port-name
   ```

14.9 The Spanning Tree Algorithm

14.9.1 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.

14.9.2 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.

14.9.3 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.

14.10 Changing the Spanning Tree Algorithm on the DECNIS

14.10.1 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.

14.10.2 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.

14.10.3 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.

14.11 Root Bridge and Designated Bridges

14.11.1 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.

14.11.2 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.

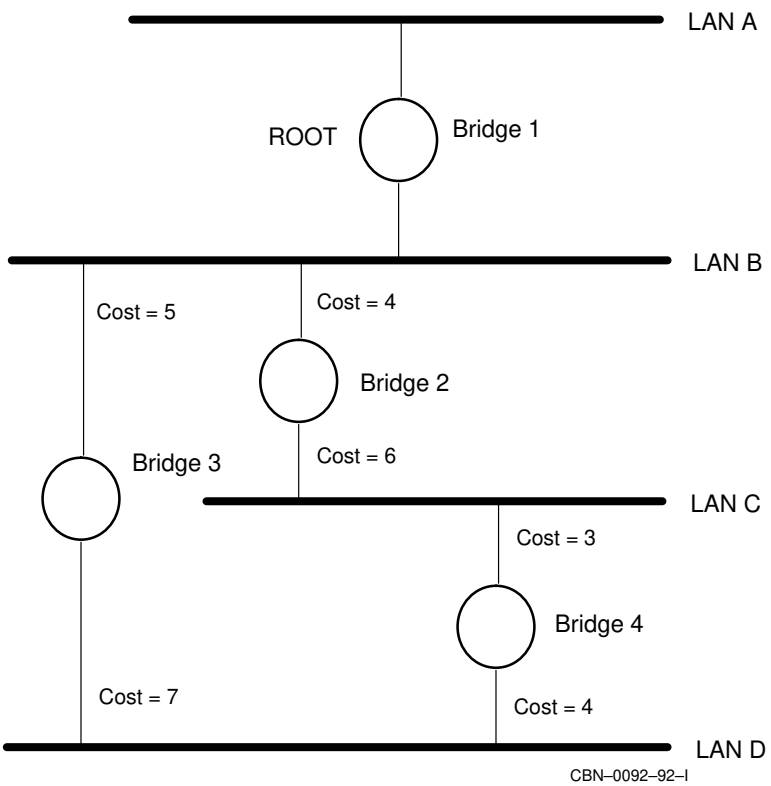
14.11.3 Example

Figure 14–4 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 14–4 Extended LAN Showing Path Costs



14.12 Changing Bridge Priority

14.12.1 Introduction

In an extended LAN, the bridge with the lowest priority becomes the root bridge.

14.12.2 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.

14.13 Changing the Port Cost

14.13.1 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.

14.13.2 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.

14.14 Minimum-Sized Ethernet/802.3 Frame Compression

14.14.1 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.

14.14.2 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 14.5 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).

14.14.3 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.

14.14.4 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.

14.14.5 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.

14.15 Bridging and DECnet Phase IV Addresses

14.15.1 External Bridges and DECnet Phase IV Addresses

If the DECNIS has been given a Phase IV address (see Section 9.2), and it has any LAN routing circuits, you may not be able to put a bridge in parallel with it.

14.15.2 Example

Figure 14–5 illustrates the situation.

14.15.3 Reason

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 9.3.

If you then try to bridge these LANs using an external (non-DECNIS) bridge, the result would be an extended LAN with two stations having the same address, which is a misconfiguration.

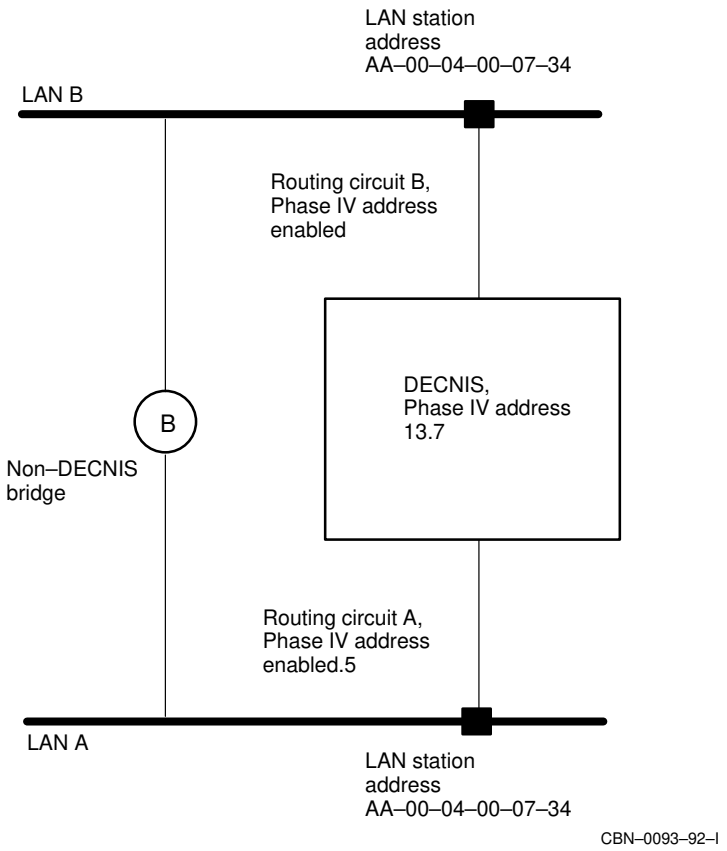
14.15.4 Solutions

If you need to use the configuration shown in the above example, do one of the following:

- Replace the external bridge shown in Figure 14–5 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.

Figure 14–5 Illegal Routing and Bridging Between Two LANs



- Set the ENABLE PHASEIV ADDRESS characteristic of one of the routing circuits to FALSE:

  ```
  NCL> SET ROUTING CIRCUIT circuit-name ENABLE PHASEIV ADDRESS FALSE
  ```

  This will prevent the LAN station address on this circuit from becoming the Phase IV style LAN address.

14.16 Remote Bridging and Loopback

You should not use loopback testing at the physical level on HDLC or PPP data links that are being used for remote bridging. That is, 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.

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 data links do not report the fact that the line is in loopback mode.

14.17 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

15

Bridge Filtering on the DECNIS

15.1 Introduction

15.1.1 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

15.1.2 NCL Commands

All tasks performed using NCL commands assume that you have already logged on to a suitable host system on the network and started NCL, and have set default to the DECNIS as follows:

```
NCL> SET NCL DEFAULT ENTITY NODE decnis
```

15.1.3 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

15.2 Bridge Filtering

15.2.1 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

15.2.2 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.

15.2.3 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.

15.3 Frame and Protocol Types

15.3.1 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)

15.3.2 Protocol Types

The protocol type of a particular frame is determined by its protocol identifier field.

15.3.3 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.

15.3.4 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.

15.4 DECNIS Filtering Process

15.4.1 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 15–1, the same checks are then carried out at the bridge port on which the frame is to be forwarded.

15.4.2 Flowchart: Bridge Filtering

Figure 15–1 illustrates bridge filtering on the DECNIS.

15.5 Management Entities Used for Protocol Filtering

15.5.1 Introduction

This section lists the entites used for controlling how the DECNIS filters or forwards frames with a particular protocol identifier.

15.5.2 Filtering Entities

Table 15–1 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 15–1 Bridge Filtering on the DECNIS

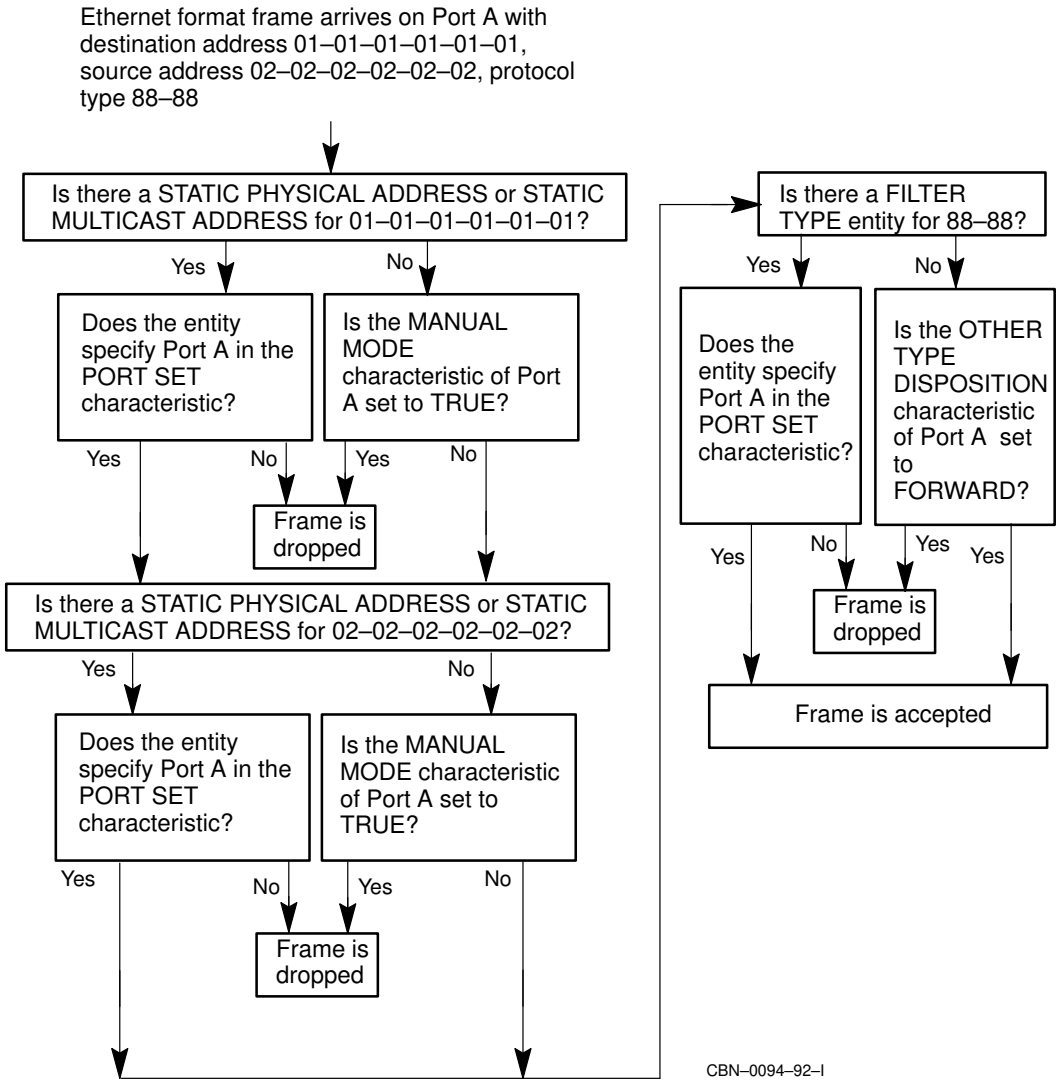


Table 15–1 Bridge Filtering Entities

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.

15.6 Permitting a Port to Bridge Only Specified Addresses

15.6.1 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.

15.6.2 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 15.10, 15.11, 15.12, and 15.13.

15.7 Permitting a Port to Bridge Only Specified Ethernet Protocol Types

15.7.1 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.

15.7.2 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 15.11 for details of how to set up FILTER TYPE entities.

15.8 Permitting a Port to Bridge Only Specified IEEE 802.2 Normal Format Protocol Types

15.8.1 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.

15.8.2 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 15.12 for details of how to set up FILTER SAP entities.

15.9 Permitting a Port to Bridge Only Specified 802.2 SNAP SAP Protocol Types

15.9.1 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.

15.9.2 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 15.13 for details of how to set up FILTER TYPE entities.

15.10 Restricting Bridging of Certain Addresses to Specific Bridge Ports

15.10.1 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).

15.10.2 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 15.14.

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 15.14.

15.11 Restricting Bridging of Certain Ethernet Protocols to Specific Bridge Ports

15.11.1 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.

15.11.2 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 15.14.

15.12 Restricting Bridging of Certain IEEE 802.2 Normal Format Protocols to Specific Bridge Ports

15.12.1 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.

15.12.2 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 15.14.

15.13 Restricting Bridging of Certain SNAP SAP Protocols to Specific Bridge Ports

15.13.1 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.

15.13.2 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 15.14.

15.14 Specifying Ports for Filtering

15.14.1 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.

15.14.2 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.

15.14.3 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.

15.14.4 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-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.

15.14.5 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.

15.14.6 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 { }
  ```

15.14.7 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
```

15.15 Locking a Destination Address to a Specific Bridge Port

15.15.1 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.

15.15.2 Example

Figure 15–2 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.

15.15.3 Procedure

1. Create a STATIC PHYSICAL ADDRESS entity for the destination address you want to lock to the port:

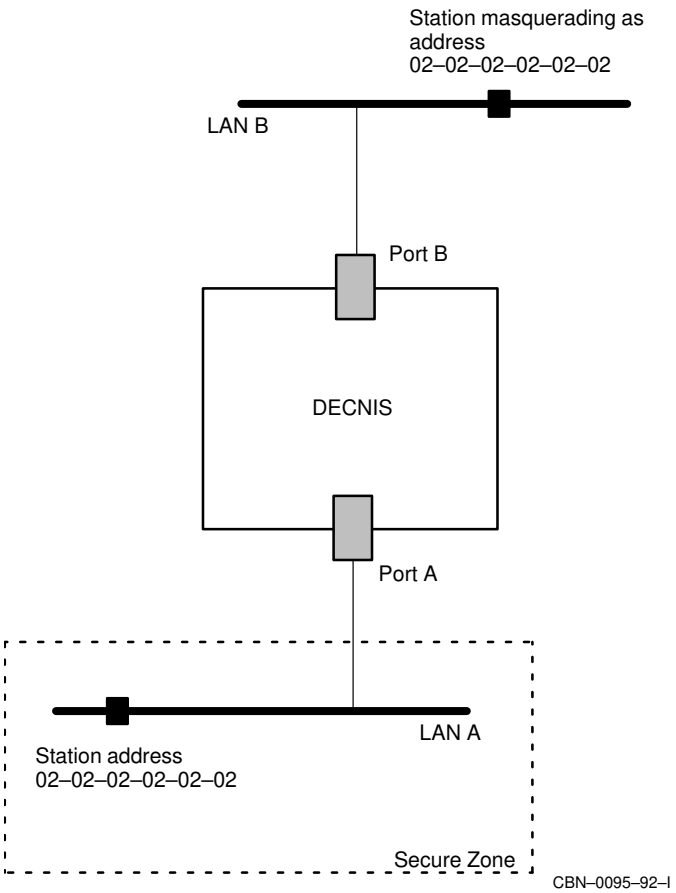
   ```
   NCL> CREATE BRIDGE STATIC PHYSICAL ADDRESS nn-nn-nn-nn-nn-nn
   ```

   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 15–2 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 15.14.

15.16 Bridging of Null Internet Frames on FDDI Ports

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 15.12.2 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}
```

15.16.1 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.

15.17 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 configurator to set up bridge filtering.

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 describes some aspects of managing the DECNIS from a DECnet–VAX Extensions system.
- Appendix H provides example NCL scripts, illustrating the NCL commands needed for the DECNIS to perform various functions.

A

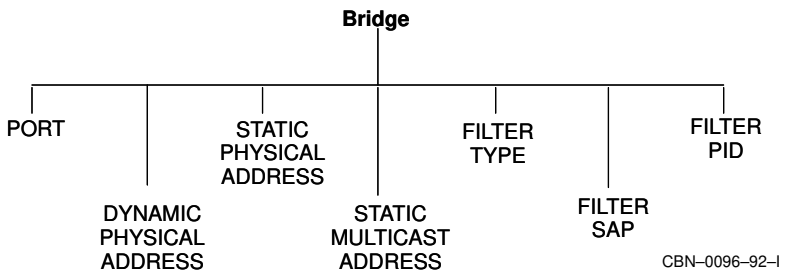
Management Modules Used by the DECNIS

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 Bridge Module

Bridging enables stations to communicate at the Data Link level; for example, bridges can connect several LANs to form an extended LAN.

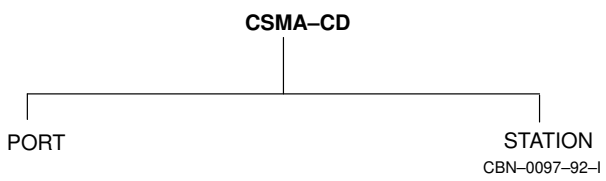
Figure A–1 Bridge Module



A.2 CSMA-CD Module

Carrier Sense, Multiple Access with Collision Detect is the Data Link protocol used by the Ethernet and ISO 8802-3 LANs.

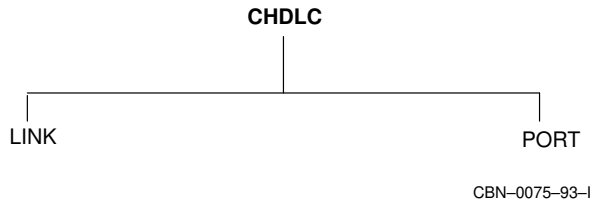
Figure A–2 CSMA-CD Module



A.3 CHDLC Module

This is a variant of the High-level Data Link Control module (see HDLC).

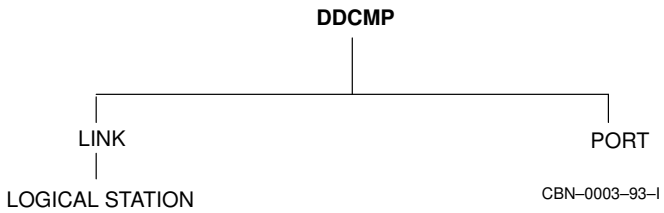
Figure A–3 CHDLC Module



A.4 DDCMP Module

The Digital Data Communications Protocol is a Digital proprietary data link protocol. The DDCMP module provides management of this protocol.

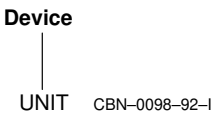
Figure A–4 DDCMP Module



A.5 Device Module

The Device module provides management of the physical devices attached to a system.

Figure A–5 Device Module



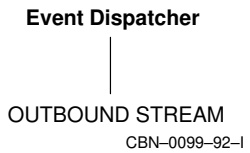
A.6 DTSS Clerk Module

The DTSS Clerk collates and synchronizes the times recorded by DTSS Servers. This module has no entities.

A.7 Event Dispatcher Module

This controls the logging of network events.

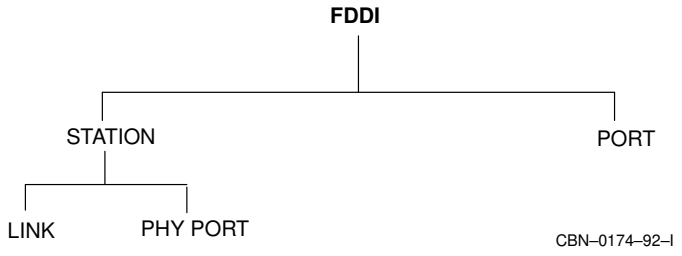
Figure A–6 Event Dispatcher Module



A.8 FDDI Module

Fiber Distributed Data Interface is the data link protocol used by fiber-optic LANs.

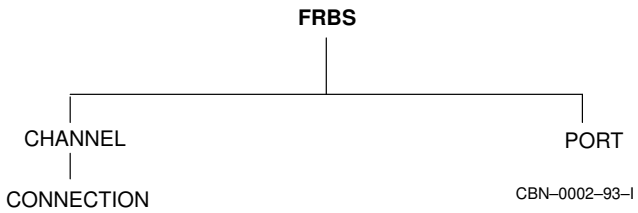
Figure A–7 FDDI Module



A.9 FRBS Module

The FRBS (Frame Relay Bearer Service) implements management of the Frame Relay functions of the DECNIS.

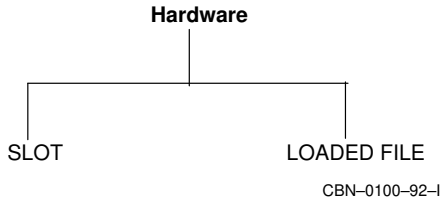
Figure A–8 FRBS Module



A.10 Hardware Module

The Hardware module provides hardware-related management of the DECNIS.

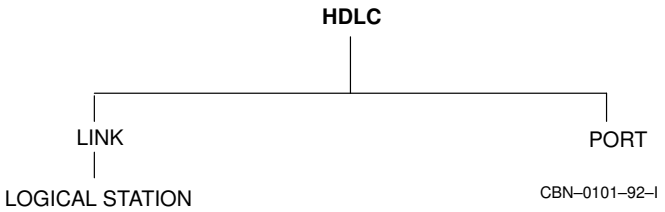
Figure A–9 Hardware Module



A.11 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.

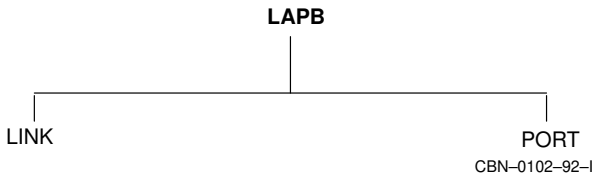
Figure A–10 HDLC Module



A.12 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.

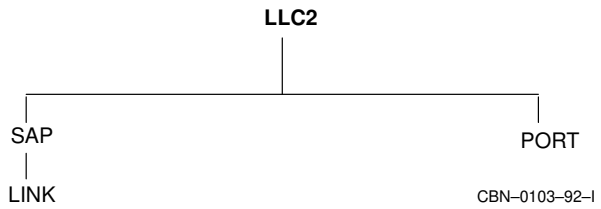
Figure A–11 LAPB Module



A.13 LLC2 Module

This module implements the LLC2 protocol that provides the ISO Connection-Oriented Network Service at the Data Link layer.

Figure A–12 LLC2 Module



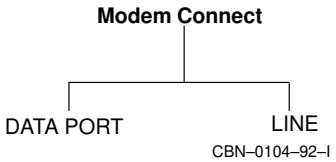
A.14 Loopback Application Module

This module controls loopback testing between two nodes at the Application layer. This module has no entities.

A.15 Modem Connect Module

This module defines the physical lines connecting two systems.

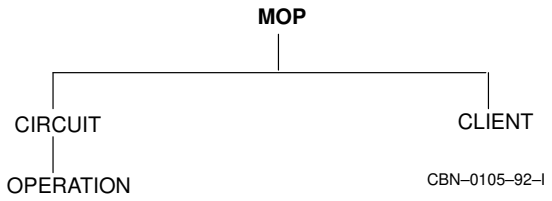
Figure A–13 Modem Connect Module



A.16 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.

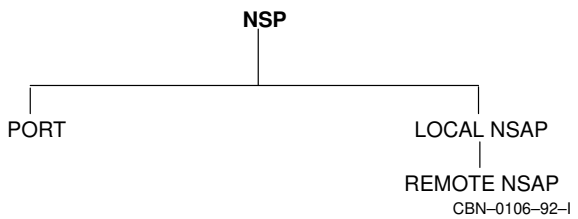
Figure A–14 MOP Module



A.17 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.

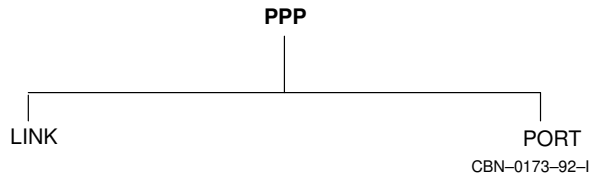
Figure A–15 NSP Module



A.18 PPP Module

The Point-to-Point Protocol is the preferred 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.

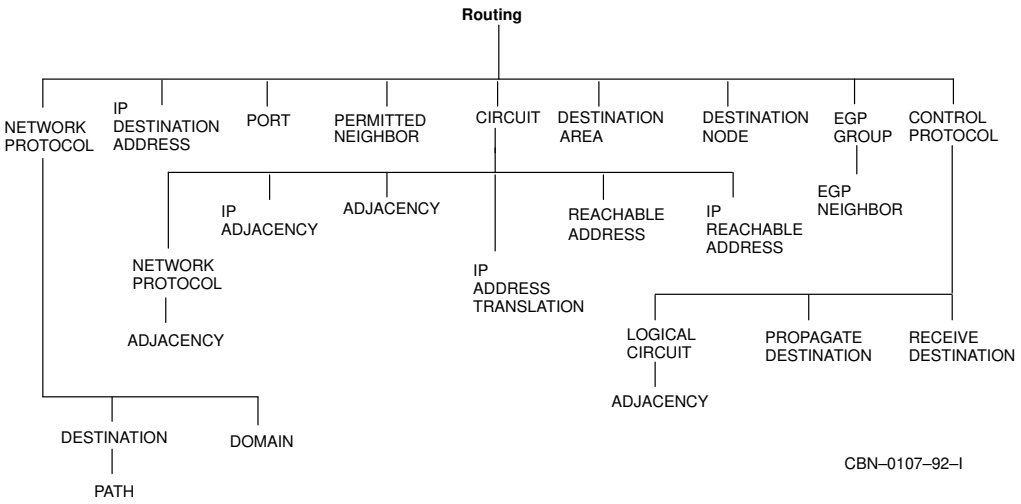
Figure A–16 PPP Module



A.19 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.

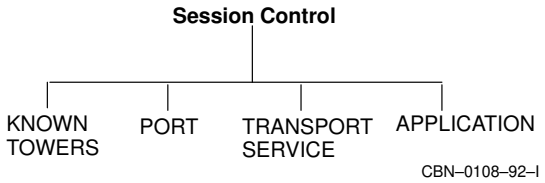
Figure A–17 Routing Module



A.20 Session Control Module

This defines the applications that receive incoming calls and make outgoing calls.

Figure A–18 Session Control Module



A.21 SNMP Module

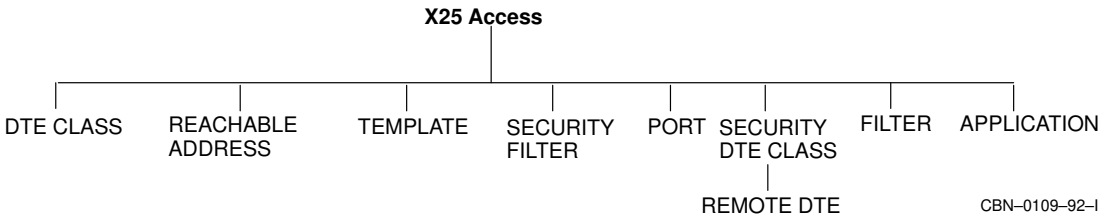
This module implements the SNMP agent on the DECNIS.

Figure A–19 SNMP Module

A.22 X25 Access Module

This defines the interface between X.25 protocols and applications.

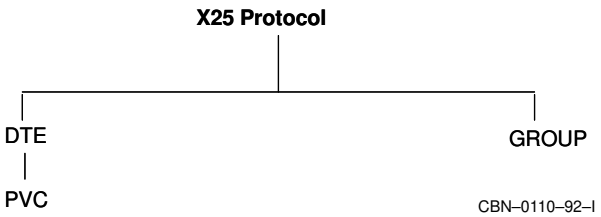
Figure A–20 X25 Access Module



A.23 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.

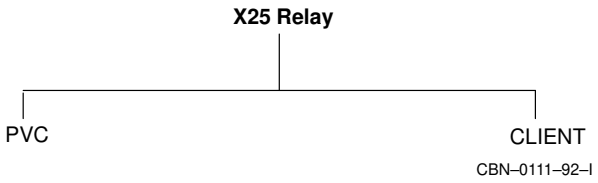
Figure A–21 X25 Protocol Module



A.24 X25 Relay Module

This defines how an X.25 call is switched from one DTE to another.

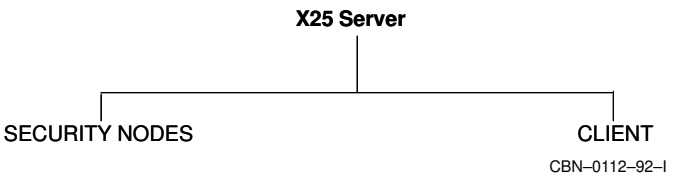
Figure A–22 X25 Relay Module



A.25 X25 Server Module

This defines how a Gateway system communicates with a Client system.

Figure A–23 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 routing characteristics, refer to the *DECnet /OSI Routing Overview*. For descriptions of the other characteristics, refer to the NCL online help and 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 gives information on such issues as 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.
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 Bridge Module

This section refers to the Bridge module.

B.1.1 BRIDGE Characteristics

Table B–1 shows the BRIDGE characteristics.

Table B–1 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.1.2 BRIDGE PORT Characteristics

Table B–2 shows the BRIDGE PORT characteristics.

Table B–2 BRIDGE PORT Characteristics

Keyword	Class	Syntax	Default Value
DATA LINK ENTITY	S	Local entity name	–
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/D	Range 1–255	–
PORT PRIORITY	S	Range 0–255	128
TYPE	R	LOCAL/REMOTE	LOCAL

B.1.3 BRIDGE STATIC PHYSICAL ADDRESS Characteristics

Table B–3 shows the BRIDGE STATIC PHYSICAL ADDRESS characteristics.

Table B–3 BRIDGE STATIC PHYSICAL ADDRESS Characteristics

Keyword	Class	Syntax	Default Value
MANUAL DESTINATION PORT	S	Port name	–
PORT SET	S	Set of port names	{ }

B.1.4 BRIDGE STATIC MULTICAST ADDRESS Characteristics

Table B–4 shows the BRIDGE STATIC MULTICAST ADDRESS characteristics.

Table B–4 BRIDGE STATIC MULTICAST ADDRESS Characteristics

Keyword	Class	Syntax	Default Value
PORT SET	S	Set of port names	{ }

B.1.5 BRIDGE FILTER SAP Characteristics

Table B–5 shows the BRIDGE FILTER SAP characteristics.

Table B–5 BRIDGE FILTER SAP Characteristics

Keyword	Class	Syntax	Default Value
PORT SET	S	Set of port names	{ }

B.1.6 BRIDGE FILTER PID Characteristics

Table B–6 shows the BRIDGE FILTER PID characteristics.

Table B–6 BRIDGE FILTER PID Characteristics

Keyword	Class	Syntax	Default Value
PORT SET	S	Set of port names	{ }

B.1.7 BRIDGE FILTER TYPE Characteristics

Table B–7 shows the BRIDGE FILTER TYPE characteristics.

Table B–7 BRIDGE FILTER TYPE Characteristics

Keyword	Class	Syntax	Default Value
PORT SET	S	Set of port names	{ }

B.2 CHDLC Module

This section refers to the CHDLC module.

B.2.1 CHDLC Characteristic

Table B–8 shows the CHDLC characteristic.

Table B–8 CHDLC Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.2.2 CHDLC LINK Characteristics

Table B–9 shows the CHDLC LINK characteristics.

Table B–9 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	R	Synchronous or Asynchronous	–

B.3 CSMA-CD Module

This section refers to the CSMA-CD module.

B.3.1 CSMA-CD Characteristics

Table B–10 shows the CSMA-CD characteristics.

Table B–10 CSMA-CD Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.3.2 CSMA-CD STATION Characteristics

Table B–11 shows the CSMA-CD STATION characteristics.

Table B–11 CSMA-CD STATION Characteristics

Keyword	Class	Syntax	Default Value
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	R	One of: Physical Virtual VCP Virtual PPP	–
TINYGRAM COMPRESSION	S/D	TRUE/FALSE	FALSE

B.4 DDCMP Module

This section refers to the DDCMP module.

B.4.1 DDCMP LINK Characteristics

Table B–12 shows the DDCMP LINK characteristics.

Table B–12 DDCMP LINK Characteristics

Keyword	Class	Syntax	Default Value
PHYSICAL LINE	S/D	Local entity name	–
RETRANSMIT TIMER	S	Range 1–65535	3000

B.4.2 DDCMP LOGICAL STATION Characteristics

Table B–13 shows the DDCMP LOGICAL STATION characteristics.

Table B–13 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.5 Device Module

This section refers to the Device module.

B.5.1 DEVICE Characteristics

Table B–14 shows the DEVICE characteristic.

Table B–14 DEVICE Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.5.2 DEVICE UNIT Characteristics

Table B–15 shows the DEVICE UNIT characteristics.

Table B–15 DEVICE UNIT Characteristics

Keyword	Class	Syntax	Default Value
AUTO LOAD	S	TRUE/FALSE	TRUE
DEVICE NAME	R	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.6 Event Dispatcher Module

This section refers to the Event Dispatcher module.

B.6.1 EVENT DISPATCHER OUTBOUND STREAM Characteristics

Table B–16 shows the EVENT DISPATCHER OUTBOUND STREAM characteristics.

Table B–16 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.7 FDDI Module

This section refers to the FDDI module.

B.7.1 FDDI Characteristics

Table B–17 shows the FDDI characteristics.

Table B–17 FDDI Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.7.2 FDDI STATION Characteristics

Table B–18 shows the FDDI STATION characteristics.

Table B–18 FDDI STATION Characteristics

Keyword	Class	Syntax	Default Value
COMMUNICATION PORT	R	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.7.3 FDDI STATION LINK Characteristics

Table B–19 shows the FDDI STATION LINK characteristics.

Table B–19 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.7.4 FDDI STATION PHY PORT Characteristics

Table B–20 shows the FDDI STATION PHY PORT characteristics.

Table B–20 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.8 FRBS Module

This section refers to the FRBS module, which implements the frame relay capabilities of the DECNIS.

B.8.1 FRBS Characteristics

Table B–21 shows the FRBS characteristic.

Table B–21 FRBS Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.8.2 FRBS CHANNEL Characteristics

Table B–22 shows the FRBS CHANNEL characteristics.

Table B–22 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	R	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	R	One of: Joint ANSI CCITT	–

B.8.3 FRBS CHANNEL CONNECTION Characteristic

Table B–23 shows the FRBS CHANNEL CONNECTION characteristic.

Table B–23 FRBS CHANNEL CONNECTION Characteristic

Keyword	Class	Syntax	Default Value
PREFERRED DLCI	S/D	Range 0–4294967295	0

B.9 Hardware Module

This section refers to the hardware module.

B.9.1 HARDWARE Characteristics

Table B–24 shows the HARDWARE characteristics.

Table B–24 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.9.2 HARDWARE SLOT Characteristics

Table B–25 shows the HARDWARE SLOT characteristics.

Table B–25 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.10 HDLC Module

This section refers to the HDLC module.

B.10.1 HDLC Characteristic

Table B–26 shows the HDLC characteristic.

Table B–26 HDLC Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.10.2 HDLC LINK Characteristics

Table B–27 shows the HDLC LINK characteristics.

Table B–27 HDLC LINK Characteristics

Keyword	Class	Syntax	Default Value
ACKNOWLEDGE TIMER	S	Range 1–60000	3000
HOLDBACK TIMER	S/D	Range 0–60000	10
RECEIVE BUFFERS	S/D	Range 1–128	8
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
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
RETRY MAXIMUM	S/D	Range 1–255	10
PHYSICAL LINE	S/D	Local Entity Name	–

B.11 LAPB Module

This section refers to the LAPB module.

B.11.1 LAPB Characteristics

Table B–28 shows the LAPB characteristic.

Table B–28 LAPB Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.11.2 LAPB LINK Characteristics

Table B–29 shows the LAPB LINK characteristics.

Table B–29 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	R	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.12 LLC2 Module

This section refers to the LLC2 module.

B.12.1 LLC2 Characteristic

Table B–30 shows the LLC2 characteristic.

Table B–30 LLC2 Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.12.2 LLC2 SAP Characteristic

Table B–31 shows the LLC2 SAP characteristics.

Table B–31 LLC2 SAP Characteristics

Keyword	Class	Syntax	Default Value
LAN STATION	S	Local entity name	–
LOCAL LSAP ADDRESS	S	Hex string	7E

B.12.3 LLC2 SAP Link Characteristics

Table B–32 shows the LLC2 SAP LINK characteristics.

Table B–32 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
POLL TIMER	S	Range 1–60000	3000
REJECT TIMER	S	Range 1–60000	3000

(continued on next page)

Table B–32 (Cont.) LLC2 SAP LINK Characteristics

Keyword	Class	Syntax	Default Value
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.13 Loopback Application Module

This section refers to the Loopback Application module.

B.13.1 LOOPBACK APPLICATION Characteristics

Table B–33 shows the LOOPBACK APPLICATION characteristics.

Table B–33 LOOPBACK APPLICATION Characteristics

Keyword	Class	Syntax	Default Value
MAXIMUM DATA	R	Range 1500–65534	65534
MAXIMUM MIRRORS	R	Range 0–32767	0

B.14 Modem Connect Module

This section refers to the Modem Connect module.

B.14.1 MODEM CONNECT Characteristic

Table B–34 shows the MODEM CONNECT characteristic.

Table B–34 MODEM CONNECT Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.14.2 MODEM CONNECT LINE Characteristics

Table B–35 shows the MODEM CONNECT LINE characteristics.

Table B–35 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	R	Asynchronous or Synchronous	–
COMMUNICATIONS PORT	R	Simple name	–
CONNECTION TYPE	R	Nonswitched or Switched	–
DUPLEX	R	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–35 (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 value of COMMUNICATIONS MODE characteristic
MODEM PROTOCOL TYPE	S/D	AT, DMCL, V25, OR V25BIS	V25BIS
PROFILE	R	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.15 MOP Module

This section refers to the MOP module.

B.15.1 MOP Characteristics

Table B–36 shows the MOP characteristics.

Table B–36 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.15.2 MOP CIRCUIT Characteristics

Table B–37 shows the MOP CIRCUIT characteristics.

Table B–37 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	R	Circuit type	–

B.15.3 MOP CLIENT Characteristics

Table B–38 shows the MOP CLIENT characteristics.

Table B–38 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.16 PPP Module

This section refers to the PPP module.

B.16.1 PPP Characteristics

Table B–39 shows the PPP entity characteristics.

Table B–39 PPP Characteristics

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.16.2 PPP LINK Characteristics

Table B–40 shows the PPP LINK entity characteristics.

Table B–40 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 or FALSE	FALSE
PREFER PROTOCOL FIELD COMPRESSION	S/D	TRUE or FALSE	FALSE
PREFER ADDRESS AND CONTROL FIELD COMPRESSION	R	TRUE or FALSE	FALSE

(continued on next page)

Table B–40 (Cont.) PPP LINK Characteristics

Keyword	Class	Syntax	Default Value
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 CONFIGURATION PROTOCOLS	S	One or more of: Internet Protocol OSI Network Layer DECnet Phase IV Bridging PDU	All these protocols
RETRANSMISSION TIMER	S	Range 1–255	3
TYPE	R	Synchronous or Asynchronous	–

B.17 Routing Module

This section refers to the Routing module.

B.17.1 ROUTING Characteristics

Table B–41 shows the ROUTING entity characteristics.

Table B–41 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
GENERATE CHECKSUMS	S	TRUE/FALSE	FALSE
INTEGRATED ISIS PROTOCOLS	S/D	One or more of: ISO8473 IP	Defaults to the value of the 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
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
MANUAL L2 ALGORITHM	S/D	Routing Vector, Link State	Routing Vector
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

(continued on next page)

Table B–41 (Cont.) ROUTING Characteristics

Keyword	Class	Syntax	Default Value
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 MANUAL ADJACENCIES	S/D	Min: 0	60
MAXIMUM PATH SPLITS	S/D	Range 1–4	1
MAXIMUM REACHABLE DESTINATIONS	S/D	Min: 0	200
MAXIMUM ROUTER ADJACENCIES	S/D	Min: 10	170
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

(continued on next page)

Table B–41 (Cont.) ROUTING Characteristics

Keyword	Class	Syntax	Default Value
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
PROTOCOLS	R	One or more of: IP ISO8473	–
REDIRECT HOLDING TIME	S	Range 1–65535	600
RIP METRIC CLASS	S	INTERNAL, EXTERNAL	EXTERNAL
RIP SEND LOCAL METRIC	S	Range 1–16	1
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
TYPE	R	One of: L1ROUTER L2ROUTER	–
VERSION	R	Version number	–

B.17.2 ROUTING CONTROL PROTOCOL Characteristics

Table B–42 shows the ROUTING CONTROL PROTOCOL characteristics.

Table B–42 ROUTING CONTROL PROTOCOL Characteristics

Keyword	Class	Syntax	Default Value
AUTONOMOUS SYSTEM NUMBER	S/D	Range 1–65535	0
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	{ }
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

(continued on next page)

Table B–42 (Cont.) ROUTING CONTROL PROTOCOL Characteristics

Keyword	Class	Syntax	Default Value
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
RIP MODE	S/D	Broadcast or Point-to-point	Broadcast
RIP STATE	S/D	One of: Off Receive Only Send and Receive Send Only	Send and Receive
TYPE	R	One of: EGP ISIS Level 1 ISIS Level 2 IP RIP Local OSPF General OSPF Area Static	–

B.17.3 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT Characteristics

Table B–43 shows the ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT characteristics.

Table B–43 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT Characteristics

Keyword	Class	Syntax	Default Value
CIRCUIT	S/D	Local entity name	" "
METRIC	S	Range 0–4294967295	Depends on the LOGICAL CIRCUIT 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.17.4 ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT ADJACENCY Characteristics

Table B–44 shows the ROUTING CONTROL PROTOCOL LOGICAL CIRCUIT ADJACENCY characteristics.

Table B–44 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.17.5 ROUTING CONTROL PROTOCOL PROPAGATE DESTINATION Characteristics

Table B–45 shows the ROUTING CONTROL PROTOCOL PROPAGATE DESTINATION characteristics.

Table B–45 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	Depends on type of parent CONTROL PROTOCOL entity
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	–
DOMAINS	S/D	Set of local entity names	{ }
FILTER ACTION	R	One of: Pass Block Summary	–
PREFERENCE	S/D	Range 0–255	0
SOURCE ROUTE ATTRIBUTES	R	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	{ }

B.17.6 ROUTING CONTROL PROTOCOL RECEIVE DESTINATION Characteristics

Table B–46 shows the ROUTING CONTROL PROTOCOL RECEIVE DESTINATION characteristics.

Table B–46 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	{ }
FILTER ACTION	R	Pass or Block	–
PREFERENCE	S/D	Range 0–255	0
SOURCE ROUTE ATTRIBUTES	R	One or more of: ISIS Internal ISIS External OSPF External Type 1 OSPF External Type 2 OSPF Intra Area OSPF Inter Area	–

B.17.7 ROUTING NETWORK PROTOCOL DOMAIN Characteristics

Table B–47 shows the ROUTING NETWORK PROTOCOL DOMAIN characteristics.

Table B–47 ROUTING NETWORK PROTOCOL DOMAIN Characteristics

Keyword	Class	Syntax	Default Value
IP ADDRESSES	S/D	Set of IP addresses	{ }

B.17.8 ROUTING PERMITTED NEIGHBOR Characteristics

Table B–48 shows the ROUTING PERMITTED NEIGHBOR characteristics.

Table B–48 ROUTING PERMITTED NEIGHBOR Characteristics

Keyword	Class	Syntax	Default Value
ID	R	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.17.9 ROUTING CIRCUIT Characteristics

Table B–49 shows the ROUTING CIRCUIT characteristics.

Table B–49 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
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
L1 COST	S	Range 1–63	20
L1 ROUTER PRIORITY	S	Range 1–127	64
L2 COST	S	Range 1–63	20
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

[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.

(continued on next page)

Table B–49 (Cont.) ROUTING CIRCUIT Characteristics

Keyword	Class	Syntax	Default Value
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	Set of protocols specified in the ROUTING characteristic 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
SUBNET ADDRESS	S/D	Subnet Address	{0.0.0.0, 0.0.0.0}
TEMPLATE	S/D	Simple Name	No name
TRANSMIT VERIFIER	W[2]	Octet String (0–38 characters)	Null string

[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–49 (Cont.) ROUTING CIRCUIT Characteristics

Keyword	Class	Syntax	Default Value
TYPE	R	One of: CSMA-CD DDCMP FDDI HDLC PPP 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}

B.17.10 ROUTING CIRCUIT NETWORK PROTOCOL Characteristics

Table B–50 shows the ROUTING CIRCUIT NETWORK PROTOCOL characteristics.

Table B–50 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.17.11 ROUTING CIRCUIT NETWORK PROTOCOL ADJACENCY Characteristics

Table B–51 shows the ROUTING CIRCUIT NETWORK PROTOCOL ADJACENCY characteristics.

Table B–51 ROUTING CIRCUIT NETWORK PROTOCOL ADJACENCY Characteristics

Keyword	Class	Syntax	Default Value
ENCAPSULATION IP ADRESS	S/D	IP address	0.0.0.0

B.17.12 ROUTING CIRCUIT REACHABLE ADDRESS Characteristics

Table B–52 shows the ROUTING CIRCUIT REACHABLE ADDRESS characteristics.

Table B–52 ROUTING CIRCUIT REACHABLE ADDRESS Characteristics

Keyword	Class	Syntax	Default Value
ADDRESS PREFIX	R	NSAP address	No address
COST	S/D	Range 1–1023	20
DATA FORMAT	S/D	Phase V, Phase IV	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.17.13 ROUTING EGP GROUP Characteristics

Table B–53 shows the ROUTING EGP GROUP characteristics.

Table B–53 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.17.14 ROUTING EGP GROUP EGP NEIGHBOR Characteristics

Table B–54 shows the ROUTING EGP GROUP EGP NEIGHBOR characteristics.

Table B–54 ROUTING EGP GROUP EGP NEIGHBOR Characteristics

Keyword	Class	Syntax	Default Value
CIRCUIT	R	Simple name	No circuit
IP ADDRESS	R	IP address	No address
SOURCE NETWORK	S	IP address	{0.0.0.0}

B.17.15 ROUTING CIRCUIT IP REACHABLE ADDRESS Characteristics

Table B–55 shows the ROUTING IP REACHABLE ADDRESS characteristics.

Table B–55 ROUTING CIRCUIT IP REACHABLE ADDRESS Characteristics

Keyword	Class	Syntax	Default Value
DESTINATION	R	IP address	–
DTE ADDRESSES	S/D	Set of DTE addresses	{ }
METRIC	S/D	Range 1–Maximum link cost	20
METRIC CLASS	S/D	INTERNAL, EXTERNAL	EXTERNAL
NEXT HOP	S/D	IP Address	0.0.0.0

B.18 Session Control Module

This section refers to the Session Control module.

B.18.1 Session Control Characteristics

Table B–56 shows the Session Control characteristics.

Table B–56 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.18.2 SESSION CONTROL KNOWN TOWER Characteristics

Table B–57 shows the SESSION CONTROL KNOWN TOWER characteristics.

Table B–57 SESSION CONTROL KNOWN TOWER Characteristics

Keyword	Class	Syntax	Default Value
TOWERS	S	Tower Set	{ }

B.19 SNMP Module

This section refers to the SNMP module.

B.19.1 SNMP Characteristics

Table B–58 shows the SNMP characteristics.

Table B–58 SNMP Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.19.2 SNMP COMMUNITY Characteristics

Table B–59 shows the SNMP COMMUNITY characteristics.

Table B–59 SNMP COMMUNITY Characteristics

Keyword	Class	Syntax	Default Value
PRIVATE NAME	S	Community name suffix	–
SNMP TYPE	R	One or more of: Agent Director	{Agent}
ACCESS	S	Read-only/Read-write	Read-only

B.20 X25 Access Module

This section refers to the X25 Access module. This section lists the characteristics that are set for the X25 Access module. Many of the characteristics for the X25 Access module are dependent on the profile of the network you are using. For details of these characteristics, refer to the online *Network Information*.

B.20.1 X25 ACCESS Characteristics

Table B–60 shows the X25 ACCESS characteristics.

Table B–60 X25 ACCESS Characteristics

Keyword	Class	Syntax	Default Value
MAXIMUM ACTIVE PORTS	R	Decimal number	–
VERSION	R	Version number	–

B.20.2 X25 ACCESS DTE CLASS Characteristics

Table B–61 shows the X25 ACCESS DTE CLASS characteristics.

Table B–61 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	R	LOCAL or REMOTE	–
USER	W	Octet string	–

B.20.3 X25 ACCESS FILTER Characteristics

Table B–62 shows the X25 ACCESS FILTER characteristics.

Table B–62 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.20.4 X25 ACCESS REACHABLE ADDRESS Characteristics

Table B–63 shows the X25 ACCESS REACHABLE ADDRESS characteristics.

Table B–63 X25 ACCESS REACHABLE ADDRESS Characteristics

Keyword	Class	Syntax	Default Value
ADDRESS EXTENSIONS	S	TRUE/FALSE	TRUE
ADDRESS PREFIX	R	DTE address	–
DESTINATION	S	DTE address	–
DTE CLASS	S	Simple name	–
MAPPING	S	Manual/X.121	X.121

B.20.5 X25 ACCESS SECURITY DTE CLASS REMOTE DTE Characteristics

Table B–64 shows the X25 ACCESS SECURITY DTE CLASS REMOTE DTE characteristics.

Table B–64 X25 ACCESS SECURITY DTE CLASS REMOTE DTE Characteristics

Keyword	Class	Syntax	Default Value
ACL	S	Access control list	–
REMOTE ADDRESS PREFIX	R	DTE address	–
RIGHTS IDENTIFIERS	S	Set of simple names	–

B.20.6 X25 ACCESS SECURITY FILTER Characteristics

Table B–65 shows the X25 ACCESS SECURITY FILTER characteristics.

Table B–65 X25 ACCESS SECURITY FILTER Characteristics

Keyword	Class	Syntax	Default Value
ACL	S	Access control list	–

B.20.7 X25 ACCESS TEMPLATE Characteristics

Table B–66 shows the X25 ACCESS TEMPLATE characteristics.

Table B–66 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.21 X25 Protocol Module

This section refers to the X25 Protocol module.

B.21.1 X25 PROTOCOL Characteristic

Table B–67 shows the X25 PROTOCOL characteristic.

Table B–67 X25 PROTOCOL Characteristic

Keyword	Class	Syntax	Default Value
VERSION	R	Version number	–

B.21.2 X25 PROTOCOL DTE Characteristics

Table B–68 shows the X25 PROTOCOL DTE characteristics.

Table B–68 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

(continued on next page)

Table B–68 (Cont.) X25 PROTOCOL DTE Characteristics

Keyword	Class	Syntax	Default Value
LINK SERVICE PROVIDER	S	Local entity name	–
MAXIMUM ACTIVE CIRCUITS	S	Range 1–4096	512
MAXIMUM CLEAR ATTEMPTS	S	Range 1–255	Profile-dependent
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	R	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.21.3 X25 PROTOCOL DTE PVC Characteristics

Table B–69 shows the X25 PROTOCOL DTE PVC characteristics.

Table B–69 X25 PROTOCOL DTE PVC Characteristics

Keyword	Class	Syntax	Default Value
ACL	S	Access control list	–
CHANNEL	R	Range 0–4095	–
PACKET SIZE	R	Range 16–4096	–
WINDOW SIZE	R	Range 1–127	–

B.21.4 X25 PROTOCOL GROUP Characteristics

Table B–70 shows the X25 PROTOCOL DTE PVC characteristics.

Table B–70 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.22 X25 Relay Module

This section refers to the X25 Relay module.

B.22.1 X25 RELAY CLIENT Characteristics

Table B–71 shows the X25 RELAY CLIENT characteristics.

Table B–71 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.22.2 X25 RELAY PVC Characteristics

Table B–72 shows the X25 RELAY PVC characteristics.

Table B–72 X25 RELAY PVC Characteristics

Keyword	Class	Syntax	Default Value
LOCAL PVC NAME	S	Simple name	–
REMOTE DTE CLASS	S	Simple name	–
REMOTE PVC NAME	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.23 X25 Server Module

This section refers to the X25 Server module.

B.23.1 X25 SERVER Characteristics

Table B–73 shows the X25 SERVER characteristics.

Table B–73 X25 SERVER Characteristics

Keyword	Class	Syntax	Default Value
INCOMING SESSION TEMPLATE	R	Simple name	–
MAXIMUM SESSION CONNECTIONS	R	Decimal number	–
VERSION	R	Version number	–

B.23.2 X25 SERVER CLIENT Characteristics

Table B–74 shows the X25 SERVER CLIENT characteristics.

Table B–74 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.23.3 X25 SERVER SECURITY NODES Characteristics

Table B–75 shows the X25 SERVER SECURITY NODES characteristics.

Table B–75 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 an ULTRIX or OSF/1 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 6.6 for more information.

‡For more information about manual routing, see Section 9.14.

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 6.6 for more information.

‡For more information about manual routing, see Section 9.14.

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 6.6 for more information.

‡For more information about manual routing, see Section 9.14.

Table F–4 DECNIS Uses Routing Vector at Level 1 and Routing Vector at Level 2

Algorithms Used by Remote System	Connectivity Information		
	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 6.6 for more information.

‡For more information about manual routing, see Section 9.14.

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 6.6 for more information.

‡For more information about manual routing, see Section 9.14.

G

Management from DECnet–VAX Extensions Systems

G.1 Introduction

The DECNIS documentation in general describes management of DECNIS from a DECnet/OSI system. Management of a DECNIS from a DECnet–VAX Extensions system is in some cases different from management from a DECnet/OSI system. This section describes those differences.

G.2 Node Implementation Syntax on DECnet–VAX Extensions

The information in the IMPLEMENTATION attribute of the NODE entity is not formatted correctly by NCL on a DECnet–VAX Extensions system. On these systems, the product name is displayed as a hexadecimal number: %X12 means the DECNIS 500, %X0D means the DECNIS 600.

G.3 Tower Set Syntax on DECnet–VAX Extensions

The syntax used for tower sets on DECnet–VAX Extensions systems is different from that given in Chapter 1 and Chapter 3.

To take an example from Section 1.10.4:

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

becomes

```
([%X0113], [DNA$ProtID$SessCtlV3 NUMBER=19],
[DNA$ProtID$NSP], [DNA$ProtID$RoutingV3 nsap])
```

This also applies to Section 3.2.2, and also to those sections in the online Problem Solving manual on the Bookreader that refer to tower sets.

G.4 Logging Events to DECmcc on DECnet–VAX Extensions

The syntax for setting up an outbound event stream to DECmcc on a DECnet–VAX Extensions system is different from that specified in Section 3.2.4.

For example, use the following command to log events to DECmcc running on a DECnet–VAX Extensions system with an NSAP address of 37:12345:02-00:AA-02-14-78-66-11:20.

```
NCL> SET NODE decnis EVENT DISPATCHER -
_NCL> OUTBOUND STREAM stream-name SINK ADDRESS -
_NCL>  { ([%X0152], [DNA$ProtID$SessCtlV2 NAME = MCC_EVL_SINK ], -
_NCL> [DNA$ProtID$NSP], -
_NCL> [DNA$ProtID$RoutingV3 37:12345:02-00:AA-02-14-78-66-11:20]) }
```

Note that a space is required between "MCC_EVL_SINK" and "]" to prevent a syntax error being reported.

G.5 Testing Event Logging on DECnet–VAX Extensions

DECnet–VAX Extensions NCL does not implement the event logging testing commands described in Section 3.13 in quite the way described.

1. The TEST directive, when used with an Outbound Stream, must be specified as TESTEVENT. When used with the Event Dispatcher module, it must be specified as TEST.
2. In TESTEVENT, PASS, BLOCK and IGNORE directives, the comma between the entity name and the event name must be omitted.

G.6 NCL "with" Clause

Commands such as the following will not work on DECnet–VAX Extensions systems:

```
NCL> SHOW CSMA-CD STATION * ALL IDENTIFIERS, -
_NCL> WITH COMMUNICATION PORT port-name
```

On DECnet–VAX Extensions systems, you need to omit the comma, and add an = after PORT, so this becomes:

```
NCL> SHOW CSMA-CD STATION * ALL IDENTIFIERS -
_NCL> WITH COMMUNICATION PORT = port-name
```

G.7 MOP Service Password when Loading from a DECnet–VAX Extensions System

The MOP Service Password in NCP is equivalent to the MOP Verification in NCL. However, the MOP Service Password has a different format to the MOP Verification value; the MOP Service Password is an 8-byte hexadecimal integer while the MOP Verification is an 8-byte hexadecimal string.

The MOP Service Password is translated into a MOP Verification value by padding the hexadecimal integer with 0s on the left until it contains 16 digits. The 8 bytes are then moved from the low-order into a hexadecimal string. So a MOP Service Password of 1234567890ABCDEF becomes a MOP Verification value of %XEFCDAB9078563412 and a MOP Service Password of DDD becomes a MOP Verification value of %XDD0D000000000000. To translate from MOP Verification value to MOP Service Password reverse the operation.

Digital recommends that you choose a value for the NCL MOP Verification which contains all 16 digits (to reduce the confusion that can arise from padding) and then set the MOP Verification attribute on the DECNIS. This value should be translated to an NCP MOP Service Password and tested using the following command:

```
NCP> LOAD NODE decnis SERVICE PASSWORD value
```

H

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.

H.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 6.22. Set the protocols to include both IP and OSI.

❷ Set the Phase V area address; see Section 9.7.

❸ Set the routing algorithm to link state; see Section 9.9.

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 5.4.

❼ Create and enable a CSMA/CD data link on the port provided by the LANcontroller 601 Network Interface Card; see Section 5.6.

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 6.3.

⓫ Set the Level 1 cost; see Section 9.11.

⓬ Set the Level 1 routing priority; see Section 9.12.

⓭ Set the IP address and subnet mask for the interface; see Section 7.5.

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 7.9.3.

H.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 6.22. Set the protocols to include both IP and OSI.

❷ Set the Phase V area address; see Section 9.7.

❸ Set the routing algorithm to link state; see Section 9.9.

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 5.4.

❺ Create the FDDI entity.

❻ Create an FDDI data link on the port provided by the FDDIcontroller 621 Network Interface Card; see Section 5.7.

❼ Create a MOP circuit on this FDDI data link, so that loopback testing can be carried out; see Section 1.11.

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 6.19.3.

⓭ Set the Level 1 cost; see Section 9.11.

⓮ Set the Level 1 routing priority; see Section 9.12.

⓯ Set the IP address and subnet mask for the interface; see Section 7.5.

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.

H.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 6.22. Set the protocols to include both IP and OSI.

❷ Set the Phase V area address; see Section 9.7.

❸ Set the routing algorithm to link state at both Level 1 and Level 2; see Section 9.9.

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 5.4.

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 5.9.

❾ The Test Mode signal on the line will not be monitored.

❿ Create the HDLC data link; see Section 5.9.

⓫ Tell the HDLC data link to use the line created above.

⓬ Set the frame-level window size for this link; see Section 5.11.

⓭ See Section 5.10.

⓮ Create the HDLC logical station; see Section 5.9.

⓯ Create the routing circuit that is to use the HDLC data link; see Section 6.5.

⓰ Specify the data link that the routing circuit is to use; see Section 6.5.

⓱ The routing circuit can be used for both Level 1 and Level 2 traffic; see Section 6.23.

⓲ Set the circuit costs for Level 1 and Level 2 traffic; see Section 9.11.

⓳ The adjacent system to which this HDLC routing circuit connects requires to receive a verifier of AFAFAF; see Section 6.24.4.

⓴ The DECNIS requires to receive a verifier of FAFAFA from the adjacent system; see Section 6.24.4.

㉑ Set the IP address and subnet mask for the IP interface.

H.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 H.3; the routing circuit will be set up on the WANcontroller 618 card configured in Section H.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 5.9.

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 5.17.

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 6.7.2.

⓭ Specify the data link that the routing circuit is to use; see Section 6.7.2.

⓮ The routing circuit can be used for both Level 1 and Level 2 traffic; see Section 6.23.

⓯ Set the circuit costs for Level 1 and Level 2 traffic; see Section 9.11.

⓰ The adjacent system to which this PPP routing circuit connects requires to receive a verifier of AFAFAF; see Section 6.24.4.

⓱ The DECNIS requires to receive a verifier of FAFAFA from the adjacent system; see Section 6.24.4.

⓲ Set the IP address and subnet mask for the IP interface.

H.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.

H.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 H.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 H.3; see Section 11.2.2.

❷ Set the line speed to 19.2 kbits/s.

❸ The line will take note of all the modem control signals; see Section 11.2.2.

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 11.2.2.

❻ Tell the LAPB data link to use the line created above.

❼ See Section 11.2.4 (acknowledge timer) and Section 11.2.5 (holdback timer).

❽ Set the maximum frame size to the value supplied by the PSDN; see Section 11.2.7.

❾ Set the frame-level window size to the value supplied by the PSDN; see Section 11.2.6.

❿ Create the DTE, specifying that it is to use the same profile as the LAPB link; see Section 11.2.2.

⓫ Specify the data link that the DTE is to use; see Section 11.2.2.

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 11.3.

⓭ Set the DTE address, as specified by the PSDN; see Section 11.2.2.

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 11.2.2.

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 11.2.10.2.

```
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 11.3.

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 11.5.2.

⓳ This filter will match calls with a call data value of FAFA; see Section 11.5.4.

⓴ Now create a server client to represent a Client system to which incoming calls will be passed; see Section 11.4.2.

㉑ Specify that it will use the filter created above to select calls; see Section 11.4.5.

㉒ Specify the Client system represented by the server client; see Section 11.4.4.

㉓ Define the Client system in the Known Towers database; see Section 1.9.

```
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 13.2.2. 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 13.2.2.

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 13.2.2. 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 13.2.2.

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 13.2.2.

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 13.4.2.

㉚ Create a SECURITY NODES entity to grant a rights identifier to all Client systems in the same namespace as the DECNIS; see Section 13.4.2. Note that it does not matter what rights identifier is granted to this SECURITY NODES entity.

H.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 H.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 H.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.

H.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 H.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
```

❶ Create the routing circuit, and specify its type; see Section 6.15.

❷ The data link for all X.25 circuits is the X25 ACCESS module; see Section 6.15.

❸ Allow the circuit to handle both Level 1 and Level 2 traffic; see Section 6.23.

❹ See Section 6.19.4.

❺ See Section 6.19.7.

❻ The number of SVC adjacencies is related to the number of SVCs allowed on this circuit; see Section 6.19.6.

❼ Specify the maximum size of data link message allowed on the circuit; see Section 6.19.2.

❽ Associate a filter with the circuit to select the OSI traffic for this circuit; see Section 6.14.

❾ Associate a filter with the circuit to select the IP traffic for this circuit; see Section 6.14.

❿ Associate a template with the circuit for outgoing OSI traffic on this circuit; see Section 6.12.

⓫ Associate a template with the circuit for outgoing IP traffic on this circuit; see Section 6.12.

```
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

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
```

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 6.12.3.

⓮ Specify the DTE class that will be used to make calls on this circuit. This DTE class was created in Section H.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 6.12.3.

⓱ Specify the DTE class that will be used to make calls on this circuit.

⓲ Create the filter to be used to select calls carrying OSI traffic; see Section 11.5.2.

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 11.5.2.

㉑ Set the call user data mask and value to check for a value of CC in the first octet of call data.

```
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
```

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 9.15. 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 9.15.

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 7.12.

㉖ Specify the IP address of the system to which this circuit will send IP packets; see Section 7.12.

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.

H.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 H.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 H.8).

❹ Set the IP address and subnet mask for the circuit; see Section 7.5.

❺ Create the template associated with this circuit; see Section 6.12.

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 H.6.

❾ Set up an EGP group to represent the autonomous system with which this routing circuit will connect; see Section 7.11.2.

❿ Specify the Autonomous System Number of this autonomous system; see Section 7.11.2. 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 7.11.2. Specify the circuit to be used for connecting with Autonomous System 125.

```
create routing control protocol egp-1 type egp ⓭
set routing control protocol egp-1 autonomous system number 125 ⓮
enable routing control protocol egp-1

create routing control protocol egp-1 propagate destination - ⓯
    pass_all_l1 filter action pass
set routing control protocol egp-1 propagate destination - ⓰
    pass_all_l1 address match type prefix
set routing control protocol egp-1 propagate destination - ⓱
    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 - ⓲
    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
```

⓭ 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.

⓮ Associate the control protocol with the EGP group created earlier by specifying the autonomous system number of the EGP group.

⓯ Create a propagate destination for passing IS-IS Level 1 routes into EGP.

⓰ Specify the matching as Prefix. With prefix matching, when a domain is not specified, all Level 1 IS-IS routes will be propagated.

⓱ Specify the type of route that is to be passed to EGP.

⓲ Now create a propagate destination for passing IS-IS Level 2 routes into EGP.

H.10 Configuring CONS LAN/WAN Relay Functions

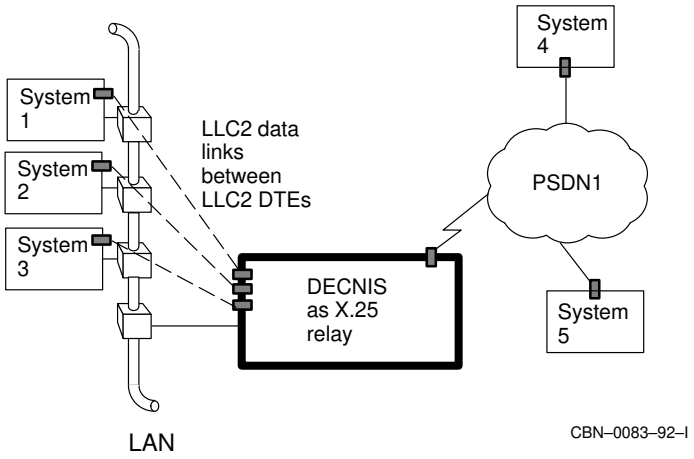
This example illustrates the NCL commands required to set up the LLC2 DTEs, and relay clients necessary to use the X.25 relay functions of the DECNIS.

In Figure H–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 H.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 H–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 H.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 H.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 H.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.

H.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 H.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 H.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 14.9.3.

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 14.12.

❸ Create a bridge port, specifying a name and number; see Section 14.5.3.

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 H.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 14.13.

❻ Specify that this bridge port is to use the HDLC data link set up in Section H.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 15.9.2

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 15.14.4.

❿ Create a STATIC PHYSICAL ADDRESS, to restrict bridging of frames bearing this address either as source or destination; see Section 15.10.1.

⓫ Use the BLOCK command to prevent a bridge port from receiving or forwarding frames bearing this address; see Section 15.14.3.

Index

A

Abbreviating IP routes, 7–38
  ROUTING SUMMARY ADDRESS, 7–38
Access Control Entry.
  See ACEs
Access Control Lists.
  See ACLs
ACEs, 13–7, 13–13, 13–21
Acknowledge timer, 5–10, 11–6
ACLs, 13–1, 13–12, 13–13, 13–21
  rights identifiers, 13–21
  wildcards, 13–21, 13–22
Adaptive routing, 7–33, 9–9
Address matching, 8–37
Address prefix, 9–10, 9–13, 9–16
Address Resolution Protocol.
  See ARP
Addressing
  area address, 9–2, 9–4
  assigning IP addresses, 7–7
  IP, 7–4
  MANUAL AREA ADDRESSES, 9–5
  NSAP addresses, 9–10, 9–16
  OSI, 9–1
  Phase IV address, 9–4
  Phase IV addresses and bridging, 14–16
  Phase IV prefix, 9–2, 9–3
  reachable addresses, 9–10, 9–16
  subnet masks, 7–5
  subnetwork address, 7–5
  system-ID, 9–2, 9–3, 9–4
ADJACENCY, 9–16
All subnets broadcast, 7–41
  data link broadcast, 7–42
  problems, 7–42
Alternate subnets, 7–22
Alternative subnet addresses, 7–44
  creating, 7–45
Announce metric, 8–23, 8–28
Announce route attributes, 8–23, 8–27
AppleTalk, 10–1
AppleTalk addresses, 10–2
AppleTalk characters, 10–7
AppleTalk name, 10–3
AppleTalk routing
  and FDDI circuits, 10–3
AppleTalk routing circuits
  creating, 10–3
  disabling, 10–4
AppleTalk zone list, 10–6
AppleTalk zones, 10–5
Area addresses, 9–2, 9–4, 9–13
Area border router, 7–11
Area restrictions, F–1
ARP, 7–42
  Proxy ARP, 7–42 to 7–44
Attributes, 1–2
Authentication, 7–10, 7–20
Autonomous system boundary router, 7–13
  set up, 7–14
Autonomous system number, 7–31
  setting, 7–28
Autonomous systems, 7–4, 7–9, 7–38
  EGP protocol, 7–28

B

Backbone router, 7–11
Backplane slots, 5–1
Backup Designated Router, 7–21
Base hardware address, 9–4
BCUGs, 11–18
  adding, 11–18
  deleting, 11–19
  members, 11–18
  remote DTE address, 11–18
  security, 13–4, 13–11
Bilateral closed user groups.
  See BCUGs
Binary CMIP, 1–6
BRIDGE entity, 14–7
Bridge filtering, 15–1 to 15–18
  BLOCK command, 15–14
  by address, 15–2
  by protocol, 15–2
  decision process, 15–4
  definition, 15–2
  entities, 14–10
  Ethernet frames, 15–7, 15–8
  Ethernet protocols, 15–7
  frame attributes, 15–2
  IEEE 802.2 Normal Format protocols, 15–8
  IEEE 802.2 SNAP SAP protocols, 15–9
  management entities, 15–4
  PORT SET, 15–14
  protocol types, 15–3
  protocols, 15–7, 15–8
  SET command, 15–14
  SHOW command, 15–16
  SNAP SAP frames, 15–9
  specifying addresses, 15–6
  specifying ports, 15–14
Bridge forwarding, 15–4
Bridge Hellos, 14–11
BRIDGE PORT entity, 14–7
Bridge ports, 14–7, 15–10
  data links, 14–8
  deleting, 14–9
Bridge ports (cont'd)
  destination address, 15–16
  filtering, 15–14
  line speed, 14–7
  local, 14–2
  monitoring, 14–9
  remote, 14–3, 14–7
Bridge security, 15–16
Bridging
  and Phase IV addresses, 14–16
  and routing, 14–6
  automatic, 15–2
  backup, 14–10
  configuring, 14–7
  definition, 14–1
  frame compression, 14–14
  frame formats, 15–3
  general, 14–1 to 14–18
  Hello messages, 14–10, 14–12
  inlinks, 14–12
  local, 14–2
  loops, 14–10
  manual, 15–2, 15–6
  over synchronous lines, 14–3
  path cost, 14–12
  protocols, 15–2
  redundancy, 14–10
  remote, 14–3
  root bridge, 14–12
  routing protocols, 14–6
  security, 15–16
  spanning tree, 14–10
Broadcast
  routing circuits, 10–12

C

Call clearing, 11–17
Call request packets, 11–15, 11–17
Call setup, 11–17
Catchall filters, 11–15
Channels, 11–5, 12–18
  PVCs, 11–17

Characteristics, 1–3
default template, E–1
values, B–1
CHDLC data links, 5–19, 5–23
adding, 5–18
CHDLC entity, 5–19, 5–23
CHDLC LINK entity, 5–19, 5–23
CHDLC routing circuits, 6–9
creating, 6–9
PHASE IV ROUTING VECTOR TIMER, 6–10
Circuit costs, 9–7
Circuit priority, 9–8
changing, 9–9
L1 ROUTER PRIORITY, 9–9
L2 ROUTER PRIORITY, 9–9
Closed user groups.
See CUGs
CMIP, 1–6
script files, 1–6, 1–7
Common Management Information Protocol.
See CMIP
Common Trace Facility.
See CTF
Communication ports, 5–1
Configuration
permanent, 1–6
Configuration parameters, 1–5
Configurator, 1–7
data files, 1–7
modify mode, 1–7
modifying permanent configuration, 1–7
Connections
CHDLC, 5–18
CSMA/CD, 5–5
deleting, 5–25
FDDI, 5–5
PPP, 5–16
VCP, 5–7
Connections to the network, 5–1 to 5–33
adding, 5–2
CHDLC over frame relay, 5–19
DDCMP, 5–12
deleting, 5–25
Connections to the network (cont'd)
disabling, 5–25
HDLC, 5–8
PPP over frame relay, 5–19
Connectivity factor, 6–33
CONS LAN/WAN relay, 12–14
CONTROL PROTOCOL entities, 7–13
Control protocol entity, 8–4
Costs
routing circuit, 9–7, 9–8
CSMA-CD entity, 5–5, 5–7
CSMA-CD STATION entity, 5–5, 5–8, 5–25
identifying from hardware port, 5–28
CSMA/CD data links, 5–5
adding, 5–5
deleting, 5–25
CSMA/CD routing circuits
creating, 6–2
deleting, 6–22
CTF
password, 4–2
security, 4–2
username, 4–2
CUGs, 11–18
adding, 11–18
deleting, 11–19
members, 11–18
Cyclic Redundancy Check, 14–15

D

Data link broadcast, 7–42
Data Link Connection Identifier, 5–20
Data links
CHDLC, 5–19, 5–23
CSMA/CD, 5–5
DDCMP, 5–12, 5–14
deleting, 5–25
FDDI, 5–5
HDLC, 5–8, 5–9
LAPB, 11–2, 11–3
PPP, 5–17, 5–23
VCP, 5–7

DCE, 11–5
DDCMP data links, 5–14
  adding, 5–12
  deleting, 5–26
  holdback timer, 5–14, 5–15
  retransmit timer, 5–14, 5–15
  transmit window, 5–15
DDCMP entity, 5–13
DDCMP LINK entity, 5–14
  identifying from modem connect line, 5–29
DDCMP routing circuits
  creating, 6–6
  security, 6–25
DDN Network Information Center, 7–28, 7–31
DEC WANrouter/Frame Relay 100/500, 5–24
DECdns, 1–11, 1–12
  clerk, 1–12
DECmcc, 1–16
DECnet Phase IV
  addressing, 9–2
  and OSI addressing, 9–2
  area address, 9–2
DECnet–VAX Extensions, G–1 to G–3
DECNIS software kit, 1–5
Default IP address, 7–7
Default route preference, 8–8
Default routes, 7–36
  advertising, 8–28
  creating, 7–37
  destination address, 7–37
  filtering, 8–20
  IP reachable addresses, 7–37
  RIP, 7–36
  static, 7–36, 7–37
Default template, E–1
Deleting connections, 5–25
Designated bridge, 14–12
Designated router, 7–15, 7–18, 7–21
Designated routers, 9–8
  backup, 7–21
  function at Level 1, 9–8
  function at Level 2, 9–8
Designated routers (cont'd)
  routing level, 9–8
DEVICE entity, 5–3
Device unit
  creating, 5–3
DEVICE UNIT entity, 5–2, 5–3
DLCI
  See Data Link Connection Identifier
DNA NEIGHBOR, 9–10
Domain entity, 8–5
Downline loading, 1–6
DTE, 11–4
DTE address, 11–4, 12–17
DTE classes, 11–2, 11–4, 11–11, 12–18
  adding, 11–11
  deleting, 11–11
  local, 11–11
  members, 11–12
  remote, 12–10
  security, 13–4, 13–8, 13–11, 13–16
  X.25 reachable addresses, 12–20
DTEs, 11–1, 11–2 to 11–11
  adding, 11–2
  characteristics, 11–8
  deleting, 11–8, 12–18
  INBOUND DTE CLASS, 11–12
  LLC2, 12–14
  packet size, 11–9
  profile, 11–4
  window size, 11–9
DTSS, 4–4
DTSS Clerk
  module, A–3
Dynamic management, 1–6

E

EGP, 7–1
EGP GROUP, 7–31
EGP NEIGHBOR, 7–31
EGP protocol, 7–28 to 7–33
  autonomous system number, 7–28, 7–31
  autonomous systems, 7–28
  configurator set up, 7–32
  example, 7–28

EGP protocol (cont'd)
filtering, 7–32
propagation, 7–32
routing level, 7–28
using, 7–28
ENABLE PHASEIV ADDRESS, 14–17
Encapsulation, 10–10
Entities, 1–2
attributes, 1–2
characteristics, 1–3
manipulating, 1–5
Ethernet
frame compression, 14–14
Event Dispatcher, A–4
EVENT DISPATCHER entity, 3–4
Event filtering, 3–7
Event filters
BLOCK command, 3–11
catchall, 3–8, 3–13
global, 3–8, 3–12
PASS command, 3–11
removing, 3–14
specific, 3–7, 3–10
TESTEVENT command, 3–15
testing, 3–15
Event logging, 3–1 to 3–16
catchall filter, 3–8, 3–13
CONNECT command, 3–5
connection timers, 3–6
DISABLE command, 3–6
disabling, 3–5
DISCONNECT command, 3–5
disconnecting, 3–5
filtering, 3–7
global filters, 3–8, 3–12
IGNORE command, 3–14
removing filters, 3–14
setting up, 3–3
SHUTDOWN command, 3–5
specific filters, 3–7, 3–10
testing filters, 3–15
Event sinks, 3–1
specifying as node name, 3–4
specifying as tower set, 3–2, 3–4
Event streams
connect retry timer, 3–6
connection timers, 3–6
disabling, 3–5
disconnect timer, 3–7
outbound, 3–1, 3–4
EXPLICIT RECEIVE VERIFICATION, 6–28
Extended LANs, 14–3, 14–10
Extended packet sequencing, 11–10
Exterior gateway protocols, 7–4

F

FDDI data links, 5–5
adding, 5–6
and AppleTalk routing, 10–3
and NetWare IPX routing, 10–8
FDDI entity, 5–6
FDDI routing circuits
creating, 6–2
deleting, 6–22
FDDI STATION entity, 5–6
identifying from hardware port, 5–28
FILTER PID entities, 15–6, 15–13
FILTER SAP entities, 15–6, 15–12
FILTER TYPE entities, 15–6, 15–11
Filtering
See Route filtering
bridges, 15–1 to 15–18
Filters, 6–23, 11–2, 11–14
adding, 11–15
and incoming calls, 11–15
catchall, 11–15
characteristics, 11–15, 11–17
deleting, 11–15
preventing access, 13–6
priority, 11–12, 11–14, 11–17
relay clients, 12–9
routing circuits, 11–14
security, 13–6, 13–7
security filters, 13–5
server clients, 11–12, 11–13, 11–14
X.25 routing circuits, 6–14, 6–16, 6–17
X.25 security, 13–4

Foreign routing domains, 9–9, 9–12
DNA protocols, 9–10
interphase links, 9–12
manual routing, 9–10
routing level, 9–10
Forwarding table, 8–1, 8–10, 8–18, 8–23
Frame compression, 14–14
Frame formats, 15–3
Ethernet, 15–3
IEEE 802.2, 15–3
802.2 SNAP SAP, 15–3
Frame relay, 5–19
DEC WANrouter/Frame Relay 100/500, 5–24
management protocol, 5–20
Permanent Virtual Circuits (PVCs), 5–20
Frame relay channel, 5–19
creating, 5–22
deleting, 5–27
Frame relay connection, 5–19
creating, 5–22
deleting, 5–27
to DEC WANrouter/Frame Relay 100/500, 5–24
Frame types
NetWare IPX, 10–10
FRBS CHANNEL CONNECTION entity, 5–22
FRBS CHANNEL entity, 5–22
FRBS entity, 5–22

G

Group number, 11–19
Groups, 11–18
adding, 11–18
deleting, 11–19
members, 11–18
type, 11–18

H

Hardware
base address, 9–4
slot, 5–1
Hardware ports, 5–28
naming, 5–1
HDLC data link
transmit buffer, 5–12
HDLC data links, 5–9
acknowledge timer, 5–10, 5–11
adding, 5–8
deleting, 5–26
holdback timer, 5–10, 5–11
preferred window size, 5–11
HDLC entity, 5–9
PREFERRED WINDOW SIZE characteristic, 5–12
HDLC LINK entity, 5–9
identifying from modem connect line, 5–29
HDLC routing circuits
creating, 6–6
deleting, 6–22
MANUAL DATA LINK SDU SIZE, 6–19
security, 6–25
Holdback timer, 5–10, 5–14, 11–6

I

INBOUND DTE CLASS, 11–11, 11–12
INCOMING LIST, 11–5, 12–18
Inlinks, 14–12, 14–14
Integrated IS–IS.
See IS–IS
Interface type, 11–9
link level, 11–9
Interior gateway protocols, 7–4
Internal router, 7–11
Internet protocol.
See IP routing
Interphase links, 9–12
creating, 9–13
example, 9–12

IP addresses, 7–4
and PPP routing circuits, 7–6
default, 7–6, 7–7
LAN circuits, 7–6
manual, 7–8
manual IP address, 7–6
point-to-point circuits, 7–6
subnet masks, 7–5
subnetwork address, 7–5
system IP address, 7–6
IP addressing
alternative subnet addresses, 7–44
assigning addresses, 7–7
neighbor IP addresses, 7–30
source address, 7–7
subnet masks, 7–7
summary addresses, 7–38
IP domain
See Domain entity
IP domains, 8–36
overlap
set up
IP gateway, 7–2
IP network
example, 7–2
IP protocols, 7–4
EGP, 7–28
exterior gateway protocols, 7–4
interior gateway protocols, 7–4
IS–IS, 7–8
OSPF, 7–9
RIP, 7–22
IP reachable addresses, 7–33, 7–37
X.25 dynamically assigned circuits, 7–35
IP routing, 7–1 to 7–47
abbreviating routes, 7–38
and OSI areas, 7–38
non-Digital systems, 7–2
routing circuits, 7–2
summary addresses, 7–38
IP routing circuits, 7–2
IP static routes, 7–33
creating, 7–35
example, 7–33
X.25 dynamically circuits, 7–35
IP-Addresses option, 7–6
IPX routing circuits
creating, 10–8
IS–IS, 7–1, 7–4, 7–8 to 7–9
filtering exceptions, 8–19
propagation exceptions, 8–29
routing algorithm, 7–8
ISO 8473, 9–15
ISO 9542, 9–9, 9–15

K

Known towers database, 1–12

L

L1 COST, 9–8
L1 ROUTER PRIORITY, 9–9
L2 COST, 9–8
L2 ROUTER PRIORITY, 9–9
LAN end systems
automatic configuration, 9–15
manual routing, 9–15
LAN segments, 14–3
LANs
extended, 14–3, 14–10
traffic, 14–3
LAPB, 11–3, A–6
LAPB data links, 11–2, 11–3
acknowledge timer, 11–6
deleting, 11–8
holdback timer, 11–6
maximum data size, 11–7
profile, 11–3
window size, 11–7
LB100 SPANNING TREE COMPATIBILITY, 14–11
Level 1 routers, 6–23
Level 1 routing traffic, 6–24
Level 2 routers, 6–23
Line profile
DATEXP, 11–3
NORMAL, 11–3

Line speed
remote bridge ports, 14–7
Lines, 5–7, 5–9, 5–13, 5–16, 5–18, 5–21
Link state routing algorithm, 9–6
Live insertion, 5–2, 5–4
LLC2 data link
deleting, 12–19
LLC2 DTEs, 12–14
changing characteristics, 12–19
creating, 12–16
deleting, 12–18
managing, 12–15
LLC2 protocol, 12–14
LNO, 1–11, 1–13
Load hosts, 1–6
OpenVMS, 1–7
security, 4–3
ULTRIX, 1–7
Loading
LOAD command, 4–3
MOP verification, 4–3
security, 4–3
Local bridging, 14–2
Local Naming Option, 1–11, 1–13
Logical circuits, 7–16
Logical station, 5–10, 5–14, 5–26
LOOP REQUESTER function, 1–16
Loopback Application
module, A–7
Loopback testing, 1–14

M

Management entities, 1–2
bridge filtering, 15–4
bridging, 14–3
routing circuits, 6–2
Management listener, 4–1
Management model, 1–2
Management modules, 1–2, A–1
Management processor, 5–1
MANUAL AREA ADDRESSES, 9–5
MANUAL DATA LINK SDU SIZE, 6–19
MANUAL DESTINATION PORT, 15–16
Manual IP address, 7–6, 7–8
MANUAL L2ONLY MODE, 6–24
Manual routing, 9–9, 9–12
adjacency, 9–16
default routes, 7–36
DNA NEIGHBOR, 9–10
end systems, 9–9
foreign routing domains, 9–10
IP, 7–33
LAN end systems, 9–15
reachable addresses, 9–10, 9–16
when to use, F–1
Master NCL script file, 1–7, 1–8
compiling, 1–8
Maximum data size, 11–7
Memory
routing, 6–29
MIB-II
system group, 4–1, 4–5
Modem Connect, A–7
Modem Connect entity, 5–7
MODEM CONNECT entity, 5–9, 5–13, 5–16, 5–18, 5–21
MODEM CONNECT LINE entity, 5–7, 5–9, 5–13, 5–16, 5–18, 5–21, 5–25, 5–26, 5–27
identifying from hardware port, 5–29
modem control, 5–9, 5–13, 5–17, 5–18, 5–21
profile, 5–7, 5–9, 5–13, 5–16, 5–18, 5–21
Modules, 1–2, A–1 to A–13
Bridge, A–1
CHDLC, A–2
CSMA/CD, A–2
DDCMP, A–3
Device, A–3
DTSS Clerk, A–3
Event Dispatcher, A–4
FDDI, A–4
FRBS, A–5
Hardware, A–5
HDLC, A–6
LAPB, A–6
LLC2, A–7
Loopback Application, A–7

Modules (cont'd)
Modem Connect, A–7
MOP, A–8
NSP, A–8
PPP, A–9
Routing, A–10
Session Control, A–11
SNMP, A–11
X25 Access, A–12
X25 Protocol, A–12
X25 Relay, A–13
X25 Server, A–13
Monitoring routing circuits, 6–22
MOP, A–8
MOP circuits, 1–14
CSMA/CD, 1–15
data links, 1–15
DDCMP, 1–15
FDDI, 1–15
HDLC, 1–15
LAPB, 1–15
LOOP REQUESTER function, 1–14
MOP verification, 4–3

N

NBMA, 7–21
NCL, 1–5
default node, 1–6
default security, 1–6
description, 1–5
documentation, 1–5
password, 4–1
script files, 1–6
security, 1–5, 4–1
starting, 1–5
username, 4–1
NCL commands
example, 1–5
NCL script files, 1–7
compiling, 1–7
converting to binary CMIP, 1–7
editing, 1–8
master, 1–7, 1–8
user, 1–8
Negotiated Neighbor IP Address, 7–6
NEGOTIATED NEIGHBOR IP ADDRESS, 7–6
Negotiation, 11–10
Neighbor IP address, 7–7, 7–30
NetBIOS, 10–11
NetWare
encapsulation, 10–10
NetBIOS, 10–11
number, 10–8
periodic routing protocol timer, 10–10
NetWare address, 10–8
NetWare IPX routing
and FDDI circuits, 10–8
NetWare IPX routing circuits
disabling, 10–9
NetWare number, 10–8
NetWare routing, 10–7
Network connections, 5–1 to 5–33
Network Control Language.
See NCL
Network interface cards
buffering, 5–12
Network Interface Cards, 5–1
adding, 5–2
configuring, 5–2
live insertion, 5–2
remote bridge ports, 14–7
Network management security, 4–1 to 4–2
Network protocol entity, 8–4
Node names, 1–12
DECdns, 1–12
translation, 1–12
Nodes database, 1–11
Nonvolatile memory, 4–3
NSAP address
Phase IV-compatible, 9–2
NSAP addresses, 1–13, 9–1, 9–10, 9–16
Digital format, 9–5
formats, 9–5
HRPF format, 9–5
OSI format, 9–5
padding digits, 9–5
Phase IV-compatible, 9–2
Phase V-style, 9–4

NSAP addresses (cont'd)
structure, 9–2
NSP, A–8
Null Internet
bridging, 15–18
NVRAM.
See Nonvolatile memory

O

OBJ_19, 4–1
OBJ_54, 4–2
Open Shortest Path First.
See OSPF
Option (3)
See IP-Addresses option
OSI addressing, 9–1
and DECnet Phase IV nodes, 9–2
OSI routing, 9–1 to 9–17
OSPF, 7–1, 7–4, 7–9
alternate subnets, 7–22
Area border router, 7–11
Area control protocol, 7–14
areas, 7–9
authentication, 7–10, 7–20
autonomous system boundary router, 7–13
backbone, 7–9
Backbone router, 7–11
configurator set up, 7–13
CONTROL PROTOCOL entities, 7–13
designated router, 7–15, 7–18, 7–21
filtering exceptions, 8–19
General control protocol, 7–13
Internal router, 7–11
introduction, 7–9
logical circuits, 7–16
minimum configuration, 7–14
multiple areas configuration, 7–16
priority, 7–15
propagation exceptions, 8–29
routing databases, 7–9
unused interfaces, 7–22
virtual links, 7–9, 7–19
OSPF interfaces
broadcast, 7–10
nonbroadcast, 7–11
point-to-point, 7–11
virtual, 7–11
OTHER PID DISPOSITION, 15–9
OTHER SAP DISPOSITION, 15–8
OTHER TYPE DISPOSITION, 15–7
Outbound event streams, 3–4
OUTGOING LIST, 11–5, 12–18

P

Packet size, 11–9
default, 11–9
maximum, 11–9
minimum, 11–9
Packet Switching Data Networks.
See PSDNs
Path costs
bridging, 14–12
Path splitting, H–3
Periodic routing protocol timer, 10–10
Permanent configuration
modifying, 1–7
Permanent virtual circuits.
See PVCs
PERMITTED NEIGHBOR entity, 6–28
Phase IV addresses, 1–13, 9–4
Phase IV prefix, 9–2
changing, 9–3
Phase IV routing algorithms, 9–6
PHASE IV ROUTING VECTOR TIMER, 6–10
Phase V area address
changing, 9–5
Point-to-point
routing circuits, 10–12
Poisoned reverse, 7–25
Pool memory, 5–1
Port cost
changing, 14–14
Port counters, 14–9

Port number, 14–7
PORT SET, 15–14
Port state, 14–9
Ports.
See Hardware ports
PPP, A–9
addressing, 7–6
PPP data links, 5–17, 5–23
adding, 5–16
deleting, 5–27
network control protocols, 5–24
to DEC WANrouter/Frame Relay 100/500, 5–24
PPP entity, 5–17, 5–23
PPP LINK entity, 5–17, 5–23
PPP routing circuits, 6–7
creating, 6–8
deleting, 6–22
network protocols, 6–7, F–1
PHASE IV ROUTING VECTOR TIMER, 6–10
Preference of routes, 8–8
Preferences
IP protocol, 8–8
PREFERRED WINDOW SIZE characteristic, 5–11
Profile
DTE, 11–4, 11–8
LAPB data link, 11–3
line, 11–3
Propagate destination entity, 8–4
Propagate destination filter, 8–22
Propagation
See Route propagation
Protocol types, 15–3
Ethernet frames, 15–3, 15–6
802.2 normal frames, 15–6
802.2 SNAP SAP frames, 15–3, 15–6
Proxy ARP, 7–42 to 7–44
configuring, 7–44
example, 7–43
PSDN, 11–4
subscription, 11–10
PSDNs, 6–11, 11–1
PVCs, 6–18, 11–17
adding, 11–17
channel number, 11–17
deleting, 11–18
packet size, 11–17
relay, 12–11
security, 13–11
window size, 11–17

R

RAP.
See Remote address prefix
Reachable addresses, 9–10, 9–12, 9–16
address prefix, 9–13
ADDRESS PREFIX, 9–10, 9–16
cost, 9–13
COST, 9–10
data format, 9–11, 9–13
inbound, 9–12
IP, 7–33, 7–35
LAN circuits, 9–11
MAPPING, 9–11
metric type, 9–11
outbound, 9–12, 9–13
remote DTE addresses, 9–11
X.25, 12–6, 12–8, 12–15, 12–19
X.25 dynamically assigned circuits, 9–11
Receive destination entity, 8–4
Receive destination filter, 8–12
RECEIVE VERIFIER, 6–27
Relay clients, 12–4, 12–7
adding, 12–8
deleting, 12–9
DTE class, 12–8
filters, 12–8, 12–9
rights identifiers, 13–17, 13–19
security, 13–16
Relay PVCs, 12–11
adding, 12–13
deleting, 12–14
security, 13–19

Remote address prefix, 13–4, 13–12
Remote bridge ports, 14–7
  spanning tree algorithm, 14–12
Remote bridging, 14–3
Remote charge access, 13–4, 13–21
REMOTE DTE, 13–4
Remote DTE classes, 12–10
  relay PVCs, 12–11
Remote DTEs
  rights identifiers, 13–6
  security, 13–11
Retransmit timer, 5–14
RFC 922, 7–42
RFC1332
  See IP-Addresses option
Rights identifiers, 13–1, 13–5, 13–11
RIP, 7–1
RIP default routes, 7–36
RIP mode, 7–26
RIP poisoned reverse, 7–25
RIP protocol, 7–22 to 7–27
  configurator set up, 7–27
  instances, 7–22
  over point-to-point circuits, 7–23
  poisoned reverse, 7–25
  propagation, 7–22
  setting up, 7–23
  unreachable routes, 7–25
RIP state, 7–25
Root bridge, 14–12, 14–14
ROOT PRIORITY, 14–14
Route advertising, 8–1, 8–5, 8–20
Route filtering, 8–1, 8–12
Route preference, 8–8
Route propagation, 8–20
  exceptions
  process, 8–23
  setting up, 8–25
Route propagation and filtering, 8–1 to 8–36
  configurator set up, 8–29
  examples, 8–31
Route types, 8–4, 8–7, 8–8
Router priority, 7–21
Routing
  adaptive, 9–9
  and bridging, 14–6
  general, 6–1 to 6–40
  IP, 7–1 to 7–47
  manual, 9–9
  NetWare, 10–7
  OSI, 9–1 to 9–17
Routing algorithms, 9–6, 9–12
  and DECnet Phase IV, 9–6
  changing, 9–7
  choosing which to run, 9–6
  IS–IS protocol, 7–8
  link state, 7–8, 9–6
  Phase IV, 9–6
  routing levels, 9–6
  routing vector, 9–6
Routing circuits
  AppleTalk, 10–3
  characteristics, 6–18
  CHDLC, 6–9
  costs, 9–7
  counters, 6–22
  CSMA/CD, 6–2
  DDCMP, 6–6
  deleting, 6–22
  FDDI, 6–2
  filters, 11–14
  for IP, 7–2
  foreign routing domains, 9–10
  HDLC, 6–6
  idle timer, 6–21
  initial minimum timer, 6–22
  IPX, 10–8
  L1 COST, 9–8
  L2 COST, 9–8
  LAN, 9–8
  maximum call attempts, 6–20
  maximum data link message size, 6–19
  maximum SVC adjacencies, 6–21
  monitoring, 6–22
  originating queue limit, 6–19
  PPP, 6–8
  priority, 9–8
  reachable addresses, 6–22

Routing circuits (cont'd)
recall timer, 6–20
security, 6–27
setting verifiers, 6–25
status, 6–22
templates, 6–12
X.25, 6–11
X25 DA, 6–15
Routing databases, 6–29
sizing, 6–29
Routing domains
foreign, 9–9, 9–10
keeping separate, 9–10
Routing level, 6–23
and circuit priority, 9–8
changing, 6–23
L1 COST, 6–25
L1 ROUTER PRIORITY, 6–25
MANUAL L2ONLY MODE, 6–24
routing algorithms, 9–6
routing circuits and Level 1 traffic, 6–24
X25 DA routing circuits, 6–15
Routing memory
adjusting, 6–29
Routing priority
designated routers, 9–8
Routing protocols
and bridging, 14–6
Routing security, 6–25
circuit types, 6–27
description, 6–25
sending verifiers, 6–29
system-level, 6–27
verifiers, 6–25
ROUTING SUMMARY ADDRESS, 7–38
Routing tables, 8–7, 8–10
Routing vector routing algorithm, 9–6
RPF
See Route propagation and filtering

S

Script files
CMIP, 1–6
NCL, 1–6
Security, 4–1 to 4–4
bridging, 15–16
DTE classes, 13–4, 13–8, 13–16
filters, 13–7
MOP verification, 4–3
NCL, 1–5
NCL, default, 1–6
relay client, 13–16
relay PVC, 13–19
routing, 6–25
SNMP, 4–4
system level, 4–1 to 4–4
X.25, 13–1 to 13–22
X.25 Client systems, 13–14
X.25 relay, 13–16
X.25 routing circuits, 6–14
SECURITY DTE CLASS, 13–4, 13–12
SECURITY FILTER, 13–5
ACL, 13–5
Security filters, 6–14
SECURITY NODES, 13–11
Client systems, 13–12
default, 13–15
Seed router, 10–4
SEQUENCE MODULUS, 5–12, 11–7
Server clients, 11–2, 11–12
adding, 11–13
deleting, 11–13
filters, 11–12, 11–13, 11–14
member system, 11–14
Session, A–11
Simple Network Management Protocol.
See SNMP
SNMP, 1–1, 2–1 to 2–10, A–11
security, 4–4
Source route attributes, 8–15, 8–23
Spanning tree, 14–10, 14–12
disabling, 14–12
remote bridge ports, 14–12

Spanning tree algorithm
autoselect, 14–11
disabling, 14–12
Hello messages, 14–11
IEEE 802.1d, 14–11
LAN Bridge 100, 14–11
remote bridge ports, 14–12
SRI International, xxxix
Static default routes, 7–36, 7–37
creating, 7–37
destination address, 7–37
Static management, 1–6
STATIC MULTICAST ADDRESS entity, 15–10
STATIC PHYSICAL ADDRESS entity, 15–10, 15–16
Static routes
default, 7–37
IP, 7–33
IP reachable addresses, 7–33
X.25 dynamically assigned circuits, 7–35, 7–37
Static routing
default routes, 7–36
Station address
Phase IV address, 14–16
Subnet addresses
alternative, 7–44
Subnet masks, 7–5, 7–7
and OSPF, 7–5
and RIP, 7–5
Subnets, 7–5
routing between, 7–44
Summary addresses, 7–38
and routing level, 7–38
creating, 7–38
System image, 4–3
System IP Address, 7–6
System security
CTF, 4–2
System-ID, 9–2, 9–3, 9–4
Phase V, 9–4

T

TEMPLATE entity, 6–13
Templates, 6–23
X.25 routing circuits, 6–16, 6–17
TESTEVENT command, 3–15
Time
setting, 4–4
Tinygram compression, 14–14
Tinygram Compression, 14–14
Tower sets, 1–12
creating, 1–12
creating with configurator, 1–12
creating with NCL, 1–13
event sinks, 3–2
example, 1–14
Towers, 1–12
structure, 1–13
Trace.
See CTF
TRANSMIT VERIFIER, 6–29
TRANSMIT WINDOW characteristic, 5–16
Tunnel circuits, 10–11

U

User NCL script files, 1–8
example, 1–11
types, 1–9

V

VCP data links, 5–7
adding, 5–7
deleting, 5–25
Verifiers, 6–25
and DECnet Phase IV, 6–27, 6–28
circuit-level, 6–25, 6–27
sending to adjacent system, 6–29
system-level, 6–25
Virtual circuits, 7–11
Virtual links, 7–9, 7–19

W

Window size, 11–7, 11–9
  default, 11–10
  maximum, 11–10
  minimum, 11–10
  negotiation, 11–10

X

X.25 ACCESS, 11–15
X.25 calls
  characteristics, 6–12, 6–14
  incoming, 11–2
  IP call user data, 6–15
  OSI call user data, 6–15
  outgoing, 11–2
X.25 circuits
  dynamically assigned, 9–9
  reachable addresses, 9–11
X.25 Client systems, 11–1, 11–12
  relay clients, 12–4
  security, 13–13, 13–14
X.25 Connector systems, 11–1
  security, 13–8
X.25 filters, 6–14
  characteristics, 6–14
  for IP routing traffic, 6–15
  for OSI routing traffic, 6–15
  security filters, 6–14
X.25 gateway, 11–1 to 11–19
X.25 networks, 11–1
X.25 point-to-point links, 11–9
X.25 reachable addresses, 12–6, 12–8, 12–15, 12–19
  creating, 12–19
  deleting, 12–20
  DTE class, 12–20
X.25 relay, 12–1 to 12–20
  security, 13–16
X.25 routing circuits, 6–11
  data link, 6–16, 6–17, 6–18
  dynamically assigned, 6–11, 6–15
  filters, 6–14, 6–15, 6–16, 6–17, 6–23
X.25 routing circuits (cont'd)
  maximum adjacencies, 6–21
  permanent, 6–18
  PVCs, 6–18
  security, 6–14, 6–25
  static, 6–11
  static incoming, 6–17
  static outgoing, 6–16
  templates, 6–12, 6–15, 6–16, 6–17, 6–23
  types, 6–11
X.25 security, 13–1 to 13–22
  Client system, 13–8
  closed system, 13–6, 13–14
  DTE classes, 13–4
  filters, 13–4
  incoming, 13–1 to 13–8
  open system, 13–6, 13–14
  outgoing, 13–8 to 13–16
  remote charge access, 13–4
  remote DTEs, 13–4
  rights identifiers, 13–1
  semiclosed system, 13–6, 13–14
X.25 templates, 6–12
  characteristics, 6–12, 6–13
  creating, 6–13
  default, E–1
  deleting, 6–13, 6–14
X25 ACCESS, 11–11
X25 ACCESS entity, 6–13
X25 ACCESS module, 6–16, 6–17, 6–18
X25 DA routing circuits
  filters for IP traffic, 6–15
  filters for OSI traffic, 6–15
X25 Protocol, A–12
X25 PROTOCOL entity, 11–4
X25 SERVER entity, 11–13