DECnet–RSX

Network Management Concepts and Procedures

Order Number: AA–EB28C–TC

October 1989

This *DECnet–RSX Network Management Concepts and Procedures* manual describes the concepts, functions, utilities, and procedures associated with DECnet–RSX network management.

Supersession/Update Information: Revised for new covers.

Operating System and Version:
RSX–11M V4.5
RSX–11S V4.5
RSX–11M–PLUS V4.3
Micro/RSX V4.3

Software Version:
DECnet–11M V4.5
DECnet–11S V4.5
DECnet–11M–PLUS V4.3
DECnet–Micro/RSX V4.3

digital™

AA–EB28C–TC
October 1989

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital or its affiliated companies.

Copyright © 1985, 1989 by Digital Equipment Corporation
All Rights Reserved.
Printed in U.S.A.

The following are trademarks of Digital Equipment Corporation:

DEC	PDP	VAX
DECmate	P/OS	VAXcluster
DECnet	Professional	VAXmate
DECUS	Rainbow	VMS
DECwriter	RSTS	VT
DIBOL	RSX	Work Processor
digital™	RT	
MASSBUS	UNIBUS	

This manual was produced by Networks and Communications Publications.

Contents

Preface

1 Network Management Overview

1.1 DECnet Interface with RSX Operating Systems 1–1
1.2 RSX–11 Packetnet System Interface 1–2
1.3 DECnet Configurations 1–2
1.3.1 Areas 1–4
1.4 Managing the Network 1–5
1.5 DECnet-RSX Databases and Utilities 1–5
1.5.1 The Permanent Database and the CFE Utility 1–6
1.5.2 The Volatile Database and the NCP Utility 1–6
1.5.3 The System Image File and the VNP Utility 1–7
1.6 Where NETGEN Ends and Network Management Begins . 1–7
1.7 Other Network Management Tools 1–9
1.7.1 Console Carrier Requester (CCR) 1–10
1.7.2 Event File Interpreter Utility (EVF) 1–10
1.7.3 Host Task Loader Utility (HLD) 1–10
1.7.4 KMX/KMY/KMV Microcode Dump Analyzer Task (KDA) 1–10
1.7.5 Network Crash Dump Analyzer (NDA) 1–11
1.7.6 Network Display Program (NTD) 1–11
1.7.7 Queue Manager Utility (QUE) 1–11
1.7.8 X.25 Trace Interpreter Task (TRI) 1–11

1.7.9 LAT Control Program (LCP) 1–12
1.7.10 Proxy Access Utility 1–12

2 Network Management Components

2.1 Nodes 2–2
2.1.1 Node Identification 2–3
2.1.1.1 Access Control 2–4
2.1.1.2 Alias Node Names 2–6
2.1.2 Specifying Access Control Verification 2–7
2.1.3 Proxy Access 2–7
2.1.4 Node Parameters 2–7
2.1.4.1 Executor Node Identification String 2–11
2.1.4.2 Executor Node Subaddresses 2–12
2.1.5 Node Counters 2–12
2.1.6 Ethernet Address of Node 2–13
2.1.6.1 Format of Ethernet Addresses 2–13
2.1.6.2 Ethernet Multicast Address Types 2–14
2.1.6.3 Ethernet Physical and Multicast Address Values 2–14
2.2 Routing Component 2–15
2.2.1 Areas 2–15
2.2.2 Types of Nodes 2–16
2.2.3 Routing Parameters 2–17
2.2.3.1 MAXIMUM ADDRESS Parameter 2–19
2.2.3.2 MAXIMUM AREAS Parameter 2–19
2.2.3.3 MAXIMUM COST and MAXIMUM HOPS Parameters 2–19
2.2.3.4 AREA MAXIMUM COST and AREA MAXIMUM HOPS Parameters 2–20
2.2.3.5 ROUTING TIMER and BROADCAST ROUTING TIMER Parameters 2–20
2.2.3.6 MAXIMUM PATH SPLITS Parameter (DECnet–11M–PLUS only) 2–21
2.2.3.7 COST Parameter (DECnet–11M and -11S Only) 2–21
2.2.3.8 LEVEL 1 COST and LEVEL 2 COST Parameters (DECnet–11M–PLUS Only) 2–23
2.2.3.9 HELLO TIMER Parameter 2–23

2.2.3.10 MAXIMUM BROADCAST ENDNODES Parameter (Ethernet Nodes Only) 2–23
2.2.3.11 Ethernet Circuit Parameters 2–23
2.3 Objects 2–24
2.3.1 Object Types 2–24
2.3.2 Object Parameters 2–25
2.3.2.1 Object UICs and Access Verification 2–26
2.3.2.2 Single Copy and Multicopy Objects 2–26
2.3.2.3 Object Names 2–26
2.4 Lines 2–27
2.4.1 Lines and Line Devices 2–27
2.4.1.1 DDCMP Line Devices 2–29
2.4.1.2 PCL Line Devices 2–29
2.4.1.3 Ethernet Line Devices 2–30
2.4.1.4 RSX–11 PSI Line Devices 2–30
2.4.2 Line Identification 2–30
2.4.3 Line Parameters 2–31
2.4.3.1 Hardware Device Parameters 2–33
2.4.3.2 Line States and Loading 2–34
2.4.3.3 RSX–11 PSI Parameters 2–34
2.4.4 Line Counters 2–35
2.5 Circuits 2–36
2.5.1 Circuit Types 2–36
2.5.1.1 DDCMP Circuits 2–36
2.5.1.2 PCL Circuits 2–36
2.5.1.3 Ethernet Circuits 2–37
2.5.1.4 DLM Circuits 2–37
2.5.1.5 RSX–11 PSI Circuits 2–37
2.5.2 Circuit Identification 2–38
2.5.2.1 DECnet Circuit Identification (DDCMP, PCL, Ethernet Circuits) 2–38
2.5.2.2 DLM Circuit Identification 2–39
2.5.2.3 RSX–11 PSI Circuit Identification 2–39
2.5.3 Circuit Parameters 2–39
2.5.3.1 Circuit States and Loading 2–41
2.5.3.2 Circuit Ownership 2–44

2.5.4 DDCMP Multipoint Circuit Parameters 2–45
2.5.4.1 Multipoint Operation 2–45
2.5.4.2 DMP and DMV Multipoint Controllers 2–46
2.5.5 DLM Circuit Parameters 2–47
2.5.5.1 Remote DTE Addresses 2–47
2.5.5.2 Re-Calls for DLM Circuits 2–48
2.5.5.3 DLM Circuit Usage 2–48
2.5.5.4 Permanent Virtual Circuit Parameters 2–49
2.5.5.5 Data Packet Control 2–49
2.5.6 RSX–11 PSI Circuit Parameters 2–50
2.5.7 Circuit Counters 2–51
2.6 Event Logging 2–51
2.6.1 Event Specification 2–53
2.6.2 Event Sources 2–54
2.6.3 Event Logger Components 2–55
2.6.4 Logging Component Names 2–55
2.6.5 Logging Sinks 2–56
2.6.6 Event Logging Component States 2–56
2.7 Counters 2–56
2.7.1 Displaying Counters 2–57
2.7.2 Zeroing Counters 2–58
2.7.3 Counter Timers and Logging Counters 2–58
2.8 Processes 2–59
2.8.1 Process Identification 2–60
2.8.2 Process States and Loading 2–63
2.8.3 Maximum Controllers and Lines 2–63
2.9 RSX–11 PSI Modules 2–64
2.9.1 X.25 Protocol Module 2–64
2.9.1.1 Local DTE-Related Parameters 2–67
2.9.1.2 Logical Channels 2–68
2.9.1.3 DTE Counters 2–68
2.9.1.4 Local DTE Line 2–69
2.9.1.5 Maximum Circuits 2–69
2.9.1.6 Local DTE States 2–69
2.9.1.7 Group-Related Parameters 2–70
2.9.1.8 Group DTE Identification 2–71
2.9.1.9 Group Number 2–71

2.9.1.10 Group Type 2–71
2.9.1.11 Protocol-Related Parameters 2–71
2.9.1.12 Packet Size 2–72
2.9.1.13 Window Size 2–72
2.9.1.14 Call Request Timer 2–73
2.9.1.15 Clear Request Timer and Maximum Clears 2–73
2.9.1.16 Reset Timer and Maximum Resets 2–73
2.9.1.17 Restart Timer and Maximum Restarts 2–74
2.9.2 X.25 Server and X.29 Server Modules 2–74
2.9.2.1 Incoming Call Handling 2–76
2.9.2.2 Destination-Related Parameters 2–77
2.9.2.3 Call Value and Call Mask (User Data Field) 2–77
2.9.2.4 Group Identification 2–78
2.9.2.5 Remote DTE Identification 2–78
2.9.2.6 Subaddresses 2–79
2.9.2.7 Object Identification 2–79
2.9.2.8 Priority 2–79
2.9.2.9 Server-Related Parameters 2–80
2.9.2.10 Maximum Circuits 2–80
2.9.2.11 X.25 Server Module Counters 2–80
2.9.2.12 X.25 Server Module States and Loading 2–81
2.9.3 X.25 Access Module 2–81
2.9.4 RSX–11 PSI Module Counters 2–82
2.10 System Component 2–82
2.10.1 System Parameters 2–82
2.10.1.1 System Buffers 2–83
2.10.1.2 System Pool 2–84
2.10.1.3 System Location 2–84

3 Operating DECnet-RSX/PSI Nodes

3.1	Controlling a Local DECnet-RSX Node	3–1
3.1.1	Starting Up a Local DECnet-RSX Node	3–2
3.1.1.1	Using the NETINS.CMD File	3–2
3.1.1.2	Using NCP Commands	3–2
3.1.1.3	Using VNP Commands	3–3
3.1.2	Loading and Turning On a Line in a Running System	3–3
3.1.3	Shutting Down DECnet-RSX	3–4
3.1.3.1	Shutting Down DECnet-RSX Circuits and Lines	3–4
3.1.3.2	Shutting Down a DECnet-RSX Node	3–4
3.2	Starting Up and Shutting Down a Local DECnet-RSX/PSI Node	3–6
3.2.1	Starting Up a Local DECnet-RSX/PSI Node	3–6
3.2.1.1	Using the NETINS.CMD File	3–6
3.2.1.2	Using NCP Commands	3–6
3.2.1.3	Using VNP Commands	3–7
3.2.2	Loading and Turning On an RSX–11 PSI Line in a Running System	3–7
3.2.3	Shutting Down RSX–11 PSI	3–8
3.2.3.1	Shutting Down RSX–11 PSI Components	3–8
3.2.3.2	Shutting Down an RSX–11 PSI Module	3–9
3.3	Monitoring DECnet-RSX/PSI Nodes	3–9
3.3.1	NCP, VNP and CFE Display Commands	3–10
3.3.1.1	Network Components	3–11
3.3.1.2	Copying NCP Display Information to a File	3–12
3.3.1.3	NCP SHOW Command Examples	3–12
3.3.2	Event Logging	3–14
3.3.3	The Network Display Program (NTD)	3–15

4 Testing the Network

4.1	Node Level Tests	4–2
4.1.1	Local-to-Remote Loopback Test	4–4
4.1.2	Loopback Tests Using a Loop Node to Specify the Circuit	4–5
4.1.2.1	Local-to-Remote Loop Node Testing	4–6
4.1.2.2	Local-to-Local Loop Node Testing	4–7
4.1.3	Local-to-Local Loopback Test	4–8
4.2	Circuit Level Tests	4–10
4.2.1	Software Loopback Test	4–11
4.2.1.1	Software Loopback Testing over Ethernet Devices	4–13
4.2.1.2	Ethernet Loopback Assistance	4–14
4.2.2	Controller Loopback Test	4–17
4.2.3	Circuit Loopback Tests	4–19
4.2.3.1	Hardware Arrangements for Circuit Loopback	4–21
4.3	X.25 Line Level Loopback Tests	4–24
4.3.1	Line Level External Loopback Tests Using a Modem	4–31
4.4	Tracing	4–32
4.5	Dumping KMX/KMY/KMV (KMS-11) Microcode	4–32

5 DECnet-RSX Host Services

5.1	Down-Line Loading	5–2
5.1.1	Down-Line System Load Operation	5–2
5.1.1.1	Target-Initiated Down-Line Loads	5–3
5.1.1.2	Operator-Initiated Down-Line Load	5–4
5.1.1.3	Load Sequence	5–6
5.1.2	Down-Line Load Requirements	5–6
5.1.2.1	Down-Line Load Bootstraps	5–7
5.1.2.2	Setting Up the Down-Line Load Devices	5–8
5.1.2.3	Setting Up the Host Circuit	5–8
5.1.3	Down-Line Load Commands	5–9
5.1.3.1	TRIGGER NODE and TRIGGER VIA Commands	5–9
5.1.3.2	LOAD NODE and LOAD VIA Commands	5–12

5.1.4 Down-Line Load Parameters 5–14
5.1.4.1 The Service Circuit Parameter 5–16
5.1.4.2 The Target Node Parameters 5–16
5.1.4.3 Down-Line Load Files 5–18
5.2 Up-Line Dumping of Memory 5–20
5.2.1 Up-Line Dump Operation 5–20
5.2.1.1 Up-Line Dump Sequence 5–20
5.2.2 Up-Line Dump Requirements 5–22
5.2.2.1 Including Up-Line Dump Support in the Satellite Node 5–22
5.2.2.2 Setting Up the Host Circuit 5–23
5.2.3 Up-Line Dump Parameters 5–23
5.3 Down-Line Loading RSX–11S Tasks 5–23
5.3.1 Down-Line Task Loading Operation 5–24
5.3.2 Down-Line Task Loading Requirements 5–25
5.3.2.1 Setting Up the Satellite System 5–25
5.3.2.2 Setting Up the Host System - Using the Host Task Loader (HLD) 5–26
5.4 Remote Console Facilities 5–28
5.4.1 CCR Operation 5–29
5.4.2 CCR Requirements 5–29
5.4.3 CCR Commands 5–30
5.4.4 CCR Special Characters 5–31
5.4.5 Sample CCR Session 5–31

6 Choosing Network Buffer Parameters

6.1 The NETCFE.CMD Command File 6–2
6.2 Allocating Memory for Buffer Space 6–2
6.3 Large Data Buffers 6–4
6.3.1 Determining LDB Size 6–4
6.3.2 Number of LDBs to Allocate 6–5
6.4 Small Data Buffers 6–7
6.5 Communications Control Buffers 6–7

7 Area Routing Guidelines

7.1	Summary of Phase IV Node Types	7–1
7.2	Area Routing Guidelines	7–2
7.3	Designing a Multiple Area Network	7–3
7.4	Converting to a Multiple Area Network	7–9
7.5	Problems in Configuring a Multiple Area Network	7–10
7.5.1	Partitioned Area Problem	7–10
7.5.2	Phase III/Phase IV Integration Problems	7–13
7.5.2.1	A Phase III Routing Node in a Phase IV Routing Path	7–15
7.5.2.2	Area Leakage Problem	7–17
7.6	Multiple Area Networks and Ethernet	7–19

8 A Component View of DECnet-RSX/PSI

8.1	The Communications Executive	8–1
8.1.1	Three Types of CEX Processes	8–2
8.1.2	The CEX Environment	8–3
8.1.2.1	Transmitting Data	8–3
8.1.2.2	Receiving Data	8–6
8.2	DECnet Communications Components	8–8
8.2.1	A Basic DECnet-RSX System	8–8
8.2.2	DECnet Physical Communications Devices	8–10
8.2.3	The Routing Control Processor	8–10
8.2.3.1	Minimum Hops/Minimum Cost Table	8–12
8.2.3.2	Reachability and Output Adjacency (ROA) Table	8–12
8.3	CEX Support Components	8–15
8.3.1	The Direct Line Access Controller	8–15
8.3.2	Network Loading Components	8–17
8.3.3	Event Logging Components	8–21
8.4	System Management Utilities	8–24
8.4.1	Modifying Network Parameters	8–24
8.4.1.1	The Network Control Program	8–25
8.4.1.2	The Configuration File Editor	8–25
8.4.1.3	The Virtual Network Processor	8–25
8.4.2	Using NCP on the Local Node	8–27

8.4.3 Using NCP on the Remote Node 8–27
8.5 System Management Support Components 8–29
8.5.1 Network Display Utility 8–29
8.5.2 Network Verification Program 8–31
8.5.3 Network Crash Dump Analyzer 8–34
8.5.4 Loopback Testing 8–35
8.5.4.1 Hardware Loopback Circuit Testing 8–36
8.5.4.2 Circuit/Line Level Loopback Testing 8–38
8.5.4.3 Node Level Loopback Testing 8–40
8.6 DECnet Utilities 8–42
8.6.1 File Access Services 8–42
8.6.1.1 The File Access Listener 8–42
8.6.1.2 The Command File/Batch File Submission Task 8–43
8.6.2 File Utilities 8–45
8.6.2.1 The Network File Transfer Utility 8–45
8.6.2.2 The File Transfer Spooler Utility 8–46
8.6.3 Terminal and Control Utilities 8–48
8.6.3.1 The Remote Terminal Utility 8–48
8.6.3.2 The Terminal Communications Utility 8–49
8.6.3.3 The Remote Task Control Utility 8–50
8.7 Satellite Support Components 8–51
8.7.1 Down-Line System Loading 8–52
8.7.2 Up-Line System Dumping 8–54
8.7.3 Down-Line Task Loading 8–56
8.8 PSI Operation 8–57
8.8.1 The PSI User Interface 8–58
8.8.2 DECnet Interface to X.25 8–60
8.8.3 X.29 Remote Terminals 8–62
8.9 Optional PSI Components 8–64
8.9.1 PSI Trace Tasks 8–65
8.9.2 KMX Microcode Tasks 8–67

A CFE, NCP, and VNP Command Summary

A.1 CFE Command Summary A–2
A.2 NCP Command Summary A–8
A.3 NCP Command Summary for RSX–11S Systems A–17
A.4 VNP Command Summary A–20

B Object Type Codes

C Network Management Event Logger Interface

D Event Class and Type Summary

D.1 Event Classes D–1
D.2 Event Message Format D–2
D.3 Network Management Layer Events D–4
D.4 Session Control Layer Events D–6
D.5 End Communications Layer Events D–6
D.6 Transport Layer Events D–7
D.7 Data Link Layer Events D–13
D.8 Physical Link Layer Events D–17
D.9 RSX System-Specific Events D–18

E Network Counter Summary

E.1 Circuit Counters E–2
E.1.1 Network Management Layer: All Circuits E–2
E.1.2 Transport Layer: All Circuits E–2
E.1.3 Data Link Layer: DDCMP Circuits E–4
E.1.4 Data Link Layer: Ethernet Circuits E–6
E.1.5 Data Link Layer: X.25 Permanent Virtual Circuits (PVCs) E–7

E.2 Line Counters E–7
E.2.1 Data Link Layer: All Devices Except DA, DMC, DMP, PCL, UNA, and QNA E–8
E.2.2 Data Link Layer: PCL Device E–8
E.2.3 Network Management Layer: All Lines Except DMC . . E–9
E.2.4 Data Link Layer: Ethernet Lines E–9
E.2.5 Data Link Layer: LAPB Lines E–12
E.3 Module Counters E–13
E.3.1 X.25 Protocol Module E–14
E.3.2 X.25/X.29 Server Modules E–16
E.4 Node Counters E–16
E.4.1 Network Management Layer E–16
E.4.2 Network Services Layer E–17
E.4.3 Executor Node Counters E–19
E.5 System Counters E–20

F Network Parameter and Counter Type Numbers

F.1 Alias Parameters (RSX System Specific) F–2
F.2 Circuit Parameters and Counters F–2
F.2.1 Circuit Parameters F–2
F.2.2 Circuit Counters F–3
F.3 Line Parameters and Counters F–6
F.3.1 Line Parameters F–6
F.3.2 Line Counters F–7
F.4 Logging Parameters and Events F–11
F.4.1 Logging Parameters F–11
F.4.2 Logging Events F–11
F.5 Node Parameters and Counters F–14
F.5.1 Node Parameters F–14
F.5.2 Node Counters F–17
F.6 Object Parameters (RSX System Specific) F–18
F.7 Process Parameters (RSX System Specific) F–18
F.8 System Parameters and Counters (RSX System Specific) . F–19
F.8.1 System Parameters F–19
F.8.2 System Counters F–19

F.9 X.25 Access Module Parameters F–20
F.10 X.25 Protocol Module Parameters and Counters F–20
F.10.1 X.25 Protocol Module Parameters F–20
F.10.2 X.25 Protocol Module Counters F–21
F.11 X.25/X.29 Server Module Parameters and Counters F–22
F.11.1 X.25/X.29 Server Module Parameters F–22
F.11.2 X.25/X.29 Server Module Counters F–23

G RSX–11 PSI Component States and State Transitions

Index

Examples

5–1 A Sample CCR Session 5–31
6–1 Excerpt from a Sample NETCFE.CMD File 6–2

Figures

1–1 Phase IV Configuration 1–3
1–2 Phase IV Area Configuration 1–4
1–3 Producing a Running DECnet-RSX System 1–8
2–1 Circuit and Node Path Cost Relationship 2–22
4–1 Local-to-Remote Loopback Test 4–4
4–2 Local-to-Remote Loopback Test Using a Loop Node Name . 4–6
4–3 Local-to-Local Loopback Test Using a Loop Node Name . . . 4–7
4–4 Local Loopback Test 4–9
4–5 Circuit Level Loopback Test 4–12
4–6 Loopback Test Using an Assistant Giving Full Assistance 4–15
4–7 Loopback Test Using an Assistant Giving Transmit Assistance 4–16
4–8 Loopback Test Using an Assistant Giving Receive Assistance 4–17
4–9 Controller Loopback Testing 4–18

4–10 Circuit Loopback Testing Using a Loopback Connector . . . 4–20
4–11 Hardware Loopback Arrangements 4–26
4–12 H315 Loopback Connector Wiring Diagram 4–27
4–13 H325 Loopback Connector Wiring Diagram 4–28
4–14 EIA Asynchronous Null Modem Wiring Diagram 4–29
4–15 Typical Breakout Box 4–30
4–16 External Loopback Test Using a Modem 4–31
5–1 Target-Initiated Down-Line Load 5–4
5–2 Operator-Initiated Down-Line Load 5–5
5–3 Operator-Initiated Down-Line Load over a DDCMP Circuit (NCP TRIGGER) 5–10
5–4 Operator-Initiated Down-Line Load over an Ethernet Circuit (NCP LOAD) 5–13
5–5 Up-Line Dumping a Remote Node 5–21
5–6 Down-Line Task Loading an RSX–11S Task 5–24
7–1 A Typical Company Requiring a Computer Network 7–4
7–2 The Level 2 Network 7–6
7–3 A Portion of the Level 1 Network in Area 5 7–8
7–4 A Network with a Partitioned Area Problem 7–12
7–5 A Phase III Routing Node in a Phase IV Routing Path 7–14
7–6 An Example of an Improperly Placed Phase III Node 7–16
7–7 Area Leakage with Phase III Nodes 7–18
7–8 A Multiple Area Ethernet 7–20
8–1 CEX Environment Illustrating Transmission 8–5
8–2 CEX Environment Illustrating Reception 8–7
8–3 A Basic DECnet-RSX System 8–9
8–4 Physical Communications Devices for a DECnet-RSX System 8–11
8–5 DECnet-RSX System Illustrating Routing 8–13
8–6 Hops/Cost Database 8–14
8–7 DECnet-RSX Operations with DLX 8–16
8–8 Loading the Network 8–18
8–9 Turning On the Network 8–19
8–10 Turning On the X.25 Server 8–20
8–11 Loading KMC Devices with Microcode 8–21
8–12 Event Logging Components 8–24
8–13 NCP, CFE, and VNP Operations 8–26
8–14 NCP, NICE, and Related Components 8–28

8–15 Network Display Utility 8–30
8–16 Network Verification Program 8–33
8–17 Network Crash Dump Analyzer 8–35
8–18 Hardware Loopback Circuit Testing 8–37
8–19 Circuit/Line Level Loopback Circuit Testing 8–39
8–20 Node Level Loopback Testing 8–41
8–21 Command/Batch File Submission Task 8–44
8–22 Network File Transfer 8–46
8–23 File Transfer Spooler 8–47
8–24 Remote Terminal Utility 8–49
8–25 Terminal Communications Utility 8–50
8–26 Remote Task Control 8–51
8–27 Down-Line System Loading 8–53
8–28 Up-Line System Dumping 8–55
8–29 Down-Line Task Loading 8–57
8–30 PSI User Interface 8–59
8–31 Physical Communications Devices for a PSI System 8–60
8–32 Combined DECnet-RSX/PSI System 8–61
8–33 PSI System with an X.29 Terminal 8–63
8–34 PSI Trace Tasks 8–66
8–35 KMX Microcode Tasks 8–67

Tables

2–1 Node Parameters and Their Functions 2–9
2–2 Routing Parameters 2–18
2–3 Object Types and Parameter Functions 2–25
2–4 Digital Communication Devices Supported by DECnet-RSX 2–28
2–5 Line Types and Parameter Functions 2–32
2–6 Circuit Types and Parameter Functions 2–40
2–7 Circuit/Line States and Substates 2–43
2–8 Functions of Logging Parameters 2–52
2–9 Process Parameters 2–60
2–10 DECnet-RSX Processes 2–61
2–11 Protocol Module Parameters 2–65
2–12 Server Module Parameters 2–75

2–13 System Parameters and Their Functions 2–83
4–1 Digital Communications Interface Specifications for Loopback Processing 4–22
5–1 Down-Line Load Bootstrap Support 5–7
5–2 Default Loader Files by Target Device Type 5–19
6–1 DECnet Buffers 6–3
6–2 LDBs Required by Different Device Types 6–6
8–1 Network Verification State 8–31
B–1 Object Type Codes B–1
D–1 Event Classes D–1
G–1 RSX–11 PSI Circuit States and Substates G–1
G–2 RSX–11 PSI Circuit State Transitions G–2
G–3 RSX–11 PSI Line States and Substates G–2
G–4 RSX–11 PSI Line State Transitions G–3
G–5 RSX–11 PSI DTE States and Substates G–4
G–6 RSX–11 PSI DTE State Transitions G–5
G–7 RSX–11 PSI Server Module States G–7
G–8 RSX–11 PSI Server Module State Transitions G–8

Preface

The *DECnet-RSX Network Management Concepts and Procedures* manual presents information needed to manage a DECnet-RSX node within a DECnet network. The term "DECnet-RSX" collectively refers to four DECnet products:

- DECnet–11M software, which runs on an RSX–11M operating system
- DECnet–11M–PLUS software, which runs on an RSX–11M–PLUS operating system
- DECnet-Micro/RSX software, which runs on a Micro/RSX operating system
- DECnet–11S software, which runs on an RSX–11S operating system

This manual also describes system management procedures for RSX–11 PSI systems and DECnet-RSX/PSI systems. The term "DECnet-RSX/PSI" refers to an RSX system that runs RSX–11 PSI software and DECnet-RSX software simultaneously. Both DECnet-RSX and RSX–11 PSI products conform to the Digital Network Architecture (DNA). DNA is the model for all DECnet implementations and the standard that allows different Digital operating systems to participate in the same network. The DNA model contains the Comite Consultatif International Telephonique et Telegraphique (CCITT) recommendation X.25, which defines a standard interface from a computer or terminal to a Packet Switching Data Network (PSDN). The RSX–11 PSI product implements the CCITT recommendation X.25 to enable an RSX operating system to participate in a PSDN.

Intended Audience

This manual is intended for anyone who is responsible for building, maintaining, and managing the network. In this manual, all such people are collectively referred to as the network manager.

Document Structure

This manual is divided into eight chapters and seven appendixes, which are summarized in the following list.

Chapter 1	Provides an overview of network management as it relates specifically to DECnet-RSX software and briefly introduces the various network management tools and utilities.
Chapter 2	Describes the basic DECnet-RSX components and summarizes the CFE, NCP, and VNP commands and parameters relevant to each.
Chapter 3	Describes how to start up and how to shut down a local DECnet-RSX or DECnet-RSX/PSI node. Chapter 3 also includes information on how to monitor network activity by using display commands and event-logging messages.
Chapter 4	Describes the various line and circuit level loopback tests that are provided to aid in fault isolation. Chapter 4 also describes the RSX-11 PSI Trace utility and the KMX/KMY /KMV dump analyzer.
Chapter 5	Describes down-line loading and up-line dumping of remote nodes and down-line task loading for RSX-11S remote nodes.
Chapter 6	Describes the allocation and usage of the various system buffers.

Chapter 7	Provides a general introduction to the concepts associated with networks that contain multiple areas. Describes the guidelines that must be followed when configuring a DECnet network that includes multiple areas. Also included in this chapter is information about procedures to convert a single area network to a multiple area network. A section on special configuration problems found in multiple area networks completes the chapter.
Chapter 8	Gives an introduction to the software components that are used in DECnet-RSX software. The internal structure of the DECnet-RSX software is described in an overview.
Appendix A	Provides a command summary for CFE, NCP, and VNP. Summarizes all CFE, NCP, and VNP commands and their parameters.
Appendix B	Provides object type codes by listing all valid object type codes with their process names.
Appendix C	Describes the network management event-logging monitor interface that you can use to enable a user-written program to process network events.
Appendix D	Provides a master list of all events and their class and type. Also provided with each event is a short explanation of the event.
Appendix E	Provides a list of all network counters that are maintained by DECnet software and includes a short description of each counter.
Appendix F	Lists all network parameters, counters, and counter type numbers as defined by the DNA Network Management Functional Specification.
Appendix G	Lists all states and state transition reasons for all RSX–11 PSI circuit, line, and module components.

Associated Documents

Before reading this manual, you should have a working knowledge of DECnet software and the RSX–11 operating system you are using. A prerequisite to the effective use of this manual is familiarity with the overall character of DECnet software as described in the following manuals.

- *DECnet-RSX Guide to Network Management Utilities*

This manual includes programming and user utility information including RSX–11 PSI programming facilities concerning DECnet-RSX software. See also the following manuals:

- *DECnet-RSX Guide to User Utilities*
- *DECnet-RSX Programmer's Reference Manual*
- *RSX–11 PSI User's Guide*

Network generation and postinstallation checkout procedures are described in the following manuals:

- *DECnet-RSX Network Generation and Installation Guide*
- *RSX–11 PSI Generation Guide*

Documentation for other Digital Ethernet Communications Server (DECSA) products includes:

- *Introduction to Local Area Networks*
- *Networks: Ethernet Products and Services Catalog*
- *DECnet Router Server Installation and Operation Guide*
- *DECnet Router/X.25 Gateway Installation and Operation Manual*
- *Terminal Server Software Installation Guide*
- *Terminal Server Operations Guide*

Other manuals that contain information related to the material covered in this manual are:

- *RSX–11M System Generation and Installation Guide*

- *RSX–11M Guide to Writing an I/O Driver*
- *RSX–11M/M–PLUS Task Builder Manual*
- *RSX–11M/M–PLUS System Management Guide*
- *RSX–11M/M–PLUS Batch and Queue Operations Manual*
- *RSX–11M/M–PLUS Utilities Manual*
- *Introduction to RSX PSI*
- *RSX P.S.I. (Packetnet Systems Interface) RSX–11 PSI System Manager's Guide*
- *DECnet–VAX System Manager's Guide*
- *Local Area Transport (LAT) Architecture Network Manager's Guide*

Information on using RSX–11 PSI software is contained in the documentation for the specific service.

An overview of the Digital Network Architecture (DNA) is provided in the following manual:

- *DECnet Digital Network Architecture (Phase IV) General Description*

The following functional specifications provide detailed descriptions of Phase IV DNA protocols:

- *DNA Network Management Functional Specification, Version 4.0.0*
- *DNA Network Services Protocol Functional Specification, Version 4.0.0*
- *DNA Digital Data Communications Message Protocol (DDCMP) Functional Specification, Version 4.1.0*
- *DNA Data Access Protocol (DAP) Functional Specification, Version 5.6.0*
- *DNA Maintenance Operations Functional Specification, Version 3.0.0*
- *DNA Session Control Functional Specification, Version 1.0.0*
- *The Ethernet, A Local Area Network, Data Link Layer and Physical Layer Specifications, Version 2.0.0*
- *DNA Ethernet Node Product Architecture Specification, Version 1.0.0*
- *DNA Ethernet Data Link Functional Specification, Version 1.0.0*
- *DNA Routing Layer Functional Specification, Version 2.0.0*

Acronyms

The following acronyms for DECnet-RSX and RSX–11 PSI components are used in this manual.

ACK	Positive acknowledge message
BCUG	Bilateral closed user group
CCB	Communications control buffer
CCITT	Comite Consultatif International Telephonique et Telegraphique
CCR	Console carrier requester
CCS	Console carrier server
CDA	Crash dump analyzer
CEX	Communications Executive
CFE	Configuration File Editor
CSR	Control status register
CUG	Closed user group
DAP	Data Access Protocol
DCB	Device control block
DCE	Data circuit-terminating equipment
DDCMP	Digital Data Communications Message Protocol
DDM	Device driver module
DLC	Data link control process
DLL	Down-line system loader
DLM	Data link mapping
DLX	Direct line access controller
DNA	Digital Network Architecture
DSR	Dynamic storage region
DTE	Data terminal equipment
DTR	DECnet test receiver
DTS	DECnet test sender

DUK	Dump KMX task
DUM	Up-line system dumper
ECL	End Communications layer
EPM	Ethernet Protocol Manager
EVF	Event File Interpreter utility
EVL	Event Logger
EVR	Event-logging receiver
FAL	File Access Listener
FRMR	Frame reject error
FTQ	File transfer queue manager
FTS	File Transfer Spooler utility
GRP	Group related parameters
HLD	Host Task Loader utility
ICB	Interrupt control block
KDA	KMX/KMY/KMV dump analyzer
KRB	Controller request block
LAPB	Link access procedure, Version B (CCITT recommendation for frame level protocol)
LAT	Local Area Transport
LCP	LAT Control Program
LDB	Large data buffer
LLC	Logical link control process
LUN	Logical unit number
MIR	Loopback mirror
MOP	Maintenance Operation Protocol
NAK	Negative acknowledge message
NCP	Network Control Program
NDA	Network Crash Dump Analyzer

NFT	Network File Transfer utility
NICE	Network Information and Control Exchange Protocol
N(R)	Next expected sequence number
NS	DECnet user interface pseudodevice
NTD	Network Display Program
NTDEMO	Network Display Server
NTINIT	Network initializer
NTL	Network loader
NW	RSX–11 PSI user interface pseudodevice
ODT	On-line debugging tool
PIP	Peripheral Interchange Program
PLI	RSX–11 PSI packet level interface
PSDN	Packet Switching Data Network
PSI	Packetnet System Interface
PSIPAD	Host based PAD
PVC	Permanent virtual circuit
QUE	Queue manager
RAM	Random access memory
RDB	Receive data buffer
RNR	Receive not ready

SCB	Status control block
SDB	Small data buffer
SLD	Satellite Task Loader utility
SVC	Switched virtual circuit
TDM	Time division multiplexed bus
TKB	RSX task builder
TRI	X.25 trace interpreter task
UCB	Unit control block
UFD	User file directory
UIC	User identification code
UMR	UNIBUS mapping register
URB	User request block
VNP	Virtual Network Processor
XDT	Executive debugging tool
XPT	Routing layer

Conventions

The following graphic conventions are used throughout this manual.

`Special type`	Special typeface is used for examples of system input or output.
Red ink	Red ink is used in examples to indicate user input. Black type represents system prompts or displays.
Gray shaded text	Gray shaded text in Appendix A indicates commands or parameters that are valid for RSX–11 PSI users only (including DLM).
Red shaded text	Red shaded text in Appendix A indicates commands or parameters that are valid for DECnet–11M–PLUS and DECnet-Micro/RSX systems only.

[ ]	Square brackets indicate that the enclosed data is optional. If a vertical list of options is enclosed, you can specify only one option. Do not type the brackets when you enter the command.
{ }	Braces indicate that you must choose one, and only one, of the enclosed options. Do not type the braces when you enter the command.
( )	Parentheses enclose a set of options that must be specified together or not at all.
Options in lists	The absence of brackets around vertical lists indicates that the items are optional.
UPPERCASE LETTERS	Indicate text that must be entered as shown. Uppercase words can be abbreviated to the first 3 or more unique characters.
italicized lowercase words	Italicized lowercase words indicate generic terms that must be replaced with specific data. In Appendix A only, NCP/VNP commands that are system specific are printed in red. All CFE commands are RSX system specific.
key	This indicates that you press a specific key. CTRL/x indicates that you should press the control key and the specified key simultaneously. Unless otherwise noted, all command lines are terminated by pressing the RETURN key.

All numbers are decimal unless otherwise noted. All Ethernet addresses are given in hexadecimal.

Example:

CLEAR	LINE *line-id*	ALL
	KNOWN LINES	COUNTER TIMER

When issuing this command, you must include a specific line ID such as DMC-0, or you must specify the KNOWN LINES keyword. Then you can specify either ALL or COUNTER TIMER. The latter is valid for RSX–11 PSI users only. You can also abbreviate the words, as shown in the following sample command:

```
NCP>CLE LIN DMC-0 ALL RET
```

1

Network Management Overview

This chapter provides a brief overview of the DECnet-RSX product, with an emphasis on topics that pertain to managing local and remote DECnet nodes. This overview describes the kinds of network configurations in which DECnet-RSX nodes can participate and introduces the databases and utilities you use to control a node's performance.

1.1 DECnet Interface with RSX Operating Systems

"DECnet" is the collective name for the software and hardware products that allow various Digital operating systems to participate in a network. DECnet-RSX is the software implementation that enables an RSX–11M, RSX–11M–PLUS, Micro/RSX, or RSX–11S operating system to function as a network node. As the RSX network interface, DECnet-RSX software supports both the protocols necessary for communicating over the network and the functions necessary for configuring, controlling, and monitoring nodes. A DECnet-RSX node can communicate with other DECnet-RSX nodes in the network or with any other Digital operating system that supports DECnet.

1.2 RSX–11 Packetnet System Interface

"RSX–11 Packetnet System Interface (PSI)" is the collective name for the software and hardware products that allow RSX–11 operating systems to participate in a packet-switching environment. A Packet Switching Data Network (PSDN) consists of switching nodes (network interfaces), to which you attach your computer or terminal, connected by high-speed links. The network interfaces are called Data Circuit-Terminating Equipment (DCE). The computers or terminals attached to the PSDN are called Data Terminal Equipment (DTE) and use Digital's implementation of the X.25 protocol to communicate with the PSDN.

With the RSX–11 PSI product, you can:

- communicate directly with other X.25 DTEs (this is the X.25 equivalent of a node).
- link DECnet nodes across a PSDN (this capability, using an X.25 virtual circuit as a DECnet data link, is known as data link mapping, or DLM).
- directly attach a computer or terminal to an X.25 PSDN without using DECnet.
- enable your system to communicate with non-Digital computers that use the X.25 protocol.

1.3 DECnet Configurations

DECnet-RSX software supports connections to both local area and wide area networks:

- Local area network connections are provided by the Ethernet and its compatible hardware.
- Wide area network connections are provided by various means, including DDCMP point-to-point and multipoint devices and data link mapping to a PSDN.

Figure 1-1 is a diagram of a Phase IV configuration that includes DECnet-RSX and DECnet–VAX nodes on an Ethernet, as well as DDCMP multipoint connections and data link mapping via an X.25 PSDN to remote DECnet nodes. As the figure shows, a DECnet-RSX node can participate in a wide area network and a local area network simultaneously.

Figure 1–1: Phase IV Configuration

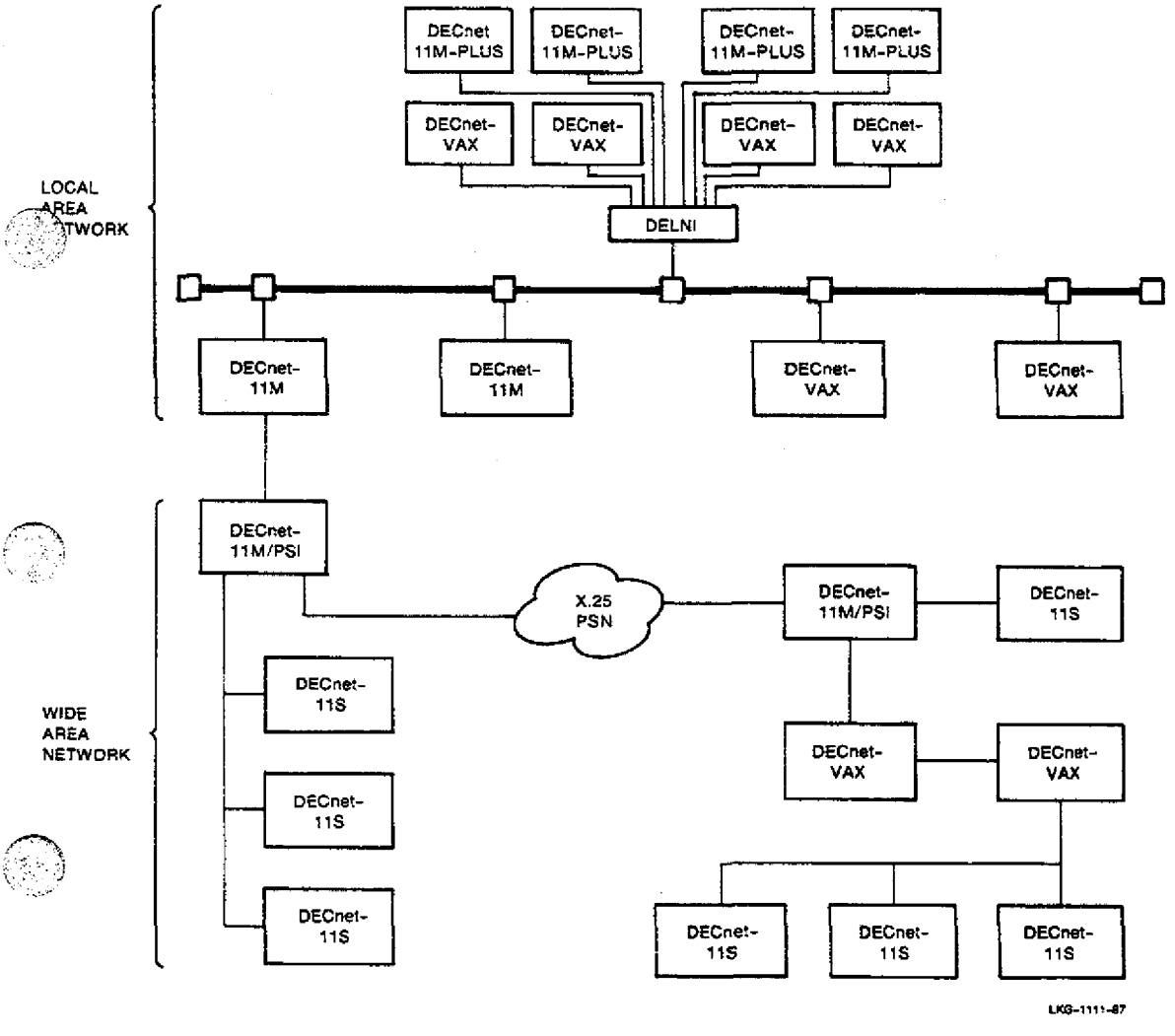
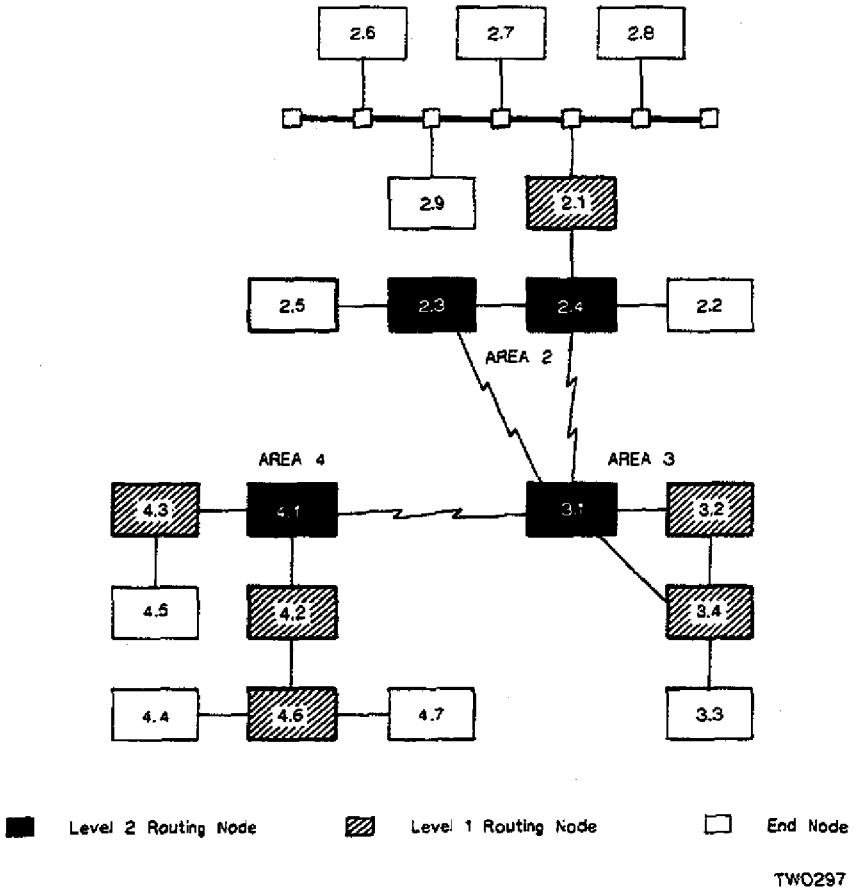


Figure 1–2: Phase IV Area Configuration



1.3.1 Areas

DECnet-RSX software also supports the area routing features of the *DNA Routing Layer Functional Specification*. These Phase IV features allow much larger network configurations than previous DNA phases. Previous phases limited network size to 255 nodes. With the support of area routing, DECnet-RSX software can now support 62 areas each with a maximum of 1023 nodes. A typical area routing example is shown in Figure 1-2. Areas allow a larger network but place more design decisions on the network manager. For more information about configuration guidelines for networks with multiple areas, see Chapter 7. In general, areas should be set up so that traffic within a single area is greater than traffic between areas.

1.4 Managing the Network

As network manager of a DECnet-RSX network, you have a number of key responsibilities, which include:

- At network generation (NETGEN), defining network components and their parameters in a central database at the local node.
- Configuring your node to ensure proper network operations in accordance with other nodes in the network. This includes defining node addresses, node names, buffer sizes, and circuit costs, all of which are parameters that affect the routing topology of the network.
- Controlling and monitoring local and remote network operations.
- Testing network hardware and software operations.
- Loading systems down line to remote nodes.

If your node includes RSX–11 PSI software, you have the following responsibilities:

- Defining RSX–11 PSI components and their parameters in the network configuration database, thus configuring the RSX–11 PSI components.
- Monitoring the operations of the RSX–11 PSI components.
- Analyzing hardware and software operations and diagnosing problems.

1.5 DECnet-RSX Databases and Utilities

Network databases provide the information that a DECnet-RSX node needs to function as part of a network. DECnet-RSX databases include:

- The permanent database
- The volatile database
- The system image file

You use a different utility to modify or define each database:

- The Configuration File Editor (CFE) modifies the permanent data base.
- The Network Control Program (NCP) modifies the volatile database.

- The Virtual Network Processor (VNP) modifies the system image file.

The complete command syntax for these three utilities is summarized in Appendix A and described in detail in the *DECnet-RSX Guide to Network Management Utilities* .

1.5.1 The Permanent Database and the CFE Utility

The permanent database is the configuration file CETAB.MAC, which is a product of NETGEN. CETAB.MAC contains the database used to load the configuration defined at NETGEN. Parameters in the permanent database are used by DECnet-RSX software whenever you load the Communications Executive, lines, circuits, and the X.25/X.29 modules for systems with an RSX-11 PSI capability. The copy of CETAB.MAC in memory after you have loaded DECnet-RSX is the volatile database. The CFE utility modifies only the CETAB.MAC permanent database. When you use CFE, the modifications to the generated system do not come into effect until the next time you load the affected part of the network software.

1.5.2 The Volatile Database and the NCP Utility

The volatile database is the DECnet-RSX system image currently in memory, and NCP is the utility you use to modify it. When you bring the network up, the volatile database includes information from the CETAB.MAC file. Changes you make using NCP remain in effect until you bring the network down and reload DECnet-RSX software or until you reboot the operating system. NCP commands allow you to load, control, and monitor the running DECnet-RSX configuration as well as to test the network hardware and software and to down-line load and up-line dump remote nodes. You use NCP commands to load the network software after you boot the operating system.

There are two versions of NCP: the full complement supported by DECnet-11M, DECnet-Micro/RSX, and DECnet-11M-PLUS software and a subset of those commands for DECnet-11S software. Most NCP commands can be executed both at your local system and at remote DECnet systems. Certain NCP commands can be executed only by privileged users. Appendix A summarizes both command sets, specifies which commands are for privileged users only, and flags commands that cannot be executed remotely. Detailed descriptions of all NCP commands and their parameters can be found in the *DECnet-RSX Guide to Network Management Utilities* .

1.5.3 The System Image File and the VNP Utility

The system image file is the RSX–11 operating system that is loaded from disk into memory when the system is booted. You use VNP, a privileged utility, to add the DECnet-RSX software to the disk system image file or to modify the DECnet-RSX software already present there. If you use VNP to load DECnet-RSX into the system image file on disk, you automatically load the DECnet software when you boot the system. VNP is comparable to the Virtual Monitor Routine (VMR), the utility you use to modify the RSX–11 operating system image.

VNP does not change the network software running in main memory. VNP changes that you make to the system image file on disk come into effect the next time the operating system is booted.

1.6 Where NETGEN Ends and Network Management Begins

The network generation procedure produces a DECnet-RSX system with or without an RSX–11 PSI capability. The system includes a configuration file, command files, and network task images. The configuration file is the CETAB.MAC file, which contains information collected during network generation. The command files include the NETINS.CMD file, which is used to load DECnet-RSX software; the NETCFE.CMD file, which stores the parameters selected during network generation; and the NETREM.CMD file, which is used to unload the network and remove network tasks. The network task images are those images selected for your system configuration. Given these products of the generation procedure, Figure 1–3 provides a step-by-step walk-through of the operations performed to produce a running DECnet-RSX system.

Figure 1–3: Producing a Running DECnet-RSX System

STEP 1 Generating a DECnet-RSX Configuration	STEP 2 Loading DECnet Software	STEP 3 Managing Local and Remote Nodes
Perform a network generation (**NETGEN**) NETGEN output: • CETAB.MAC • Command files: NETINS.CMD NETREM.CMD NETCFE.CMD • Network task images If necessary, issue **CFE** commands to modify CETAB.MAC, the permanent data base.	Load DECnet using one of three options: • Invoke **NETINS.CMD** (installs network tasks; optionally loads DECnet from CETAB.MAC and turns on local node); or • Issue **NCP** commands to load DECnet from CETAB.MAC and turn on local node; or • Issue **VNP** commands to load DECnet from CETAB.MAC into system image file; then reboot system.	Issue **NCP** commands to: • Modify the volatile data base • Test network software • Load remote systems • Perform other management tasks Issue **CFE** commands to modify CETAB.MAC before reloading DECnet. Issue **VNP** commands to modify the system image before rebooting the system.

LKG–1112–87

Though the procedures can be iterated differently within each set, the following is a sequential description of the steps:

1. **Generating the network.** Generation of the network provides the components upon which step 2, loading the network, operates. The *DECnet-RSX Network Generation and Installation Guide* describes the network

generation procedure for configuring a DECnet-RSX system with or without the RSX–11 PSI component.

You can issue CFE commands to modify the permanent configuration file (CETAB.MAC) once it has been created by the network generation procedure. This is useful for modifying configuration parameters prior to loading the network.

2. **Loading the network.** Once you have configured the system, you can load the network software into the target configuration, the actual running configuration produced by the NETGEN procedure, by invoking the NETINS.CMD command file or by using NCP or VNP commands. The end product of this step is an active DECnet-RSX configuration capable of participating in a network. At this point, you should run the postinstallation checkout procedures, as defined in the *DECnet-RSX Network Generation and Installation Guide*.

3. **Reconfiguring and tuning the running network.** Once you have loaded the network software, you can use NCP to modify the volatile database to enhance network performance and reconfigure the running network. For example, you can turn circuits on or off and enable or disable RSX–11 PSI network operations. You can also use NCP to test and monitor network software operations.

1.7 Other Network Management Tools

In addition to CFE, VNP, and NCP, DECnet-RSX software supports other network management tools to perform specific functions:

- Console carrier requester (CCR)
- Event File Interpreter utility (EVF)
- Host Task Loader utility (HLD)
- KMX/KMY/KMV microcode dump analyzer (KDA)
- Network Crash Dump Analyzer (NDA)
- Network Display Program (NTD)
- Queue Manager utility (QUE)
- Trace interpreter task (TRI)
- LAT Control Program (LCP)

- Proxy Access utility

These tools are introduced briefly in the following sections and are described in detail in the *DECnet-RSX Guide to Network Management Utilities* .

1.7.1 Console Carrier Requester (CCR)

The console carrier requester (CCR) provides remote access to console carrier services on a remote Digital Ethernet Communication Server (DECSA) system. CCR uses the DLX interface to communicate with the console carrier server (CCS) at the remote node. For more information about CCR, see Section 5.4.

1.7.2 Event File Interpreter Utility (EVF)

The Event File Interpreter (EVF) is a utility that enables you to format the event file created by the event logging facility of DECnet-RSX software. Using the DEFINE/SET LOGGING FILE command you can establish a file to receive events in a machine-readable format. The EVF utility reads this file and formats the events in a format similar to that seen on the logging console. EVF, and the formatted reports that it creates, are useful for troubleshooting network errors. Chapter 2 provides an introduction to the event logging facilities of DECnet-RSX.

1.7.3 Host Task Loader Utility (HLD)

The Host Task Loader (HLD) is a utility that enables you to down-line load tasks to a target RSX–11S system that is running DECnet software. This enables the target system to use a host node for storing task images (both overlaid and nonoverlaid) on disk and for providing disk space for checkpointing tasks. Section 5.3.2.2 describes the prerequisites for down-line loading tasks and the procedure for setting up the tables needed by the HLD utility.

1.7.4 KMX/KMY/KMV Microcode Dump Analyzer Task (KDA)

KDA is the RSX–11 PSI task which formats a KMX/KMY/KMV microcode dump on disk to produce an output listing. The microcode dump on disk is produced using the NCP KMX-DUMP command. The listing should be analyzed by a Software Services representative.

1.7.5 Network Crash Dump Analyzer (NDA)

The Network Crash Dump Analyzer (NDA) is a DECnet-RSX utility used to analyze crash dumps of RSX systems that are running DECnet-RSX software, including RSX–11 PSI systems, when a crash occurs. NDA is a nonprivileged utility that uses the network symbol table file to compile and analyze information from a crashed system image. The utility creates both a formatted file from the dump media and a listing printout. NDA is useful only if you are familiar with internal data structures and databases for DECnet-RSX and RSX–11 PSI software.

You cannot use NDA to analyze a crash dump unless the executive crash dump routine has been built into the RSX operating system during system generation. See the *RSX–11M System Generation and Installation Guide*.

1.7.6 Network Display Program (NTD)

NTD provides real-time displays of network status information such as active network tasks and resources allocated to the system or a summary of reachable nodes. The display information changes as the status changes. You can run NTD to display resource information about your node, any other Phase IV DECnet-RSX node, or any Phase III DECnet-RSX or DECnet-IAS node in the network.

1.7.7 Queue Manager Utility (QUE)

The Queue Manager utility (QUE) provides the interface to the DECnet file transfer queue. The QUE utility itself is also a feature of the RSX operating system. You use QUE, a privileged utility, to initialize the File Transfer Spooler (FTS) queue and processor and to manipulate user-queued FTS requests.

1.7.8 X.25 Trace Interpreter Task (TRI)

The trace interpreter task (TRI) is an RSX–11 PSI facility that allows you to diagnose software problems on an X.25 communication line. You can trace line messages and write them to a file. Use TRI to analyze and print information in this file. TRI is useful only if you are familiar with the X.25 level 2 and 3 protocols.

1.7.9 LAT Control Program (LCP)

LAT Control Program (LCP) commands enable you to monitor and control the LAT process on a DECnet-RSX-11M-PLUS or a DECnet-Micro/RSX node. Local Area Transport (LAT) is a communication system that permits your RSX-11M-PLUS or Micro/RSX system to access remote devices (such as printers) that are on a terminal server. With LAT, the communication between your system and the terminal servers takes place over a single shared Ethernet physical layer.

1.7.10 Proxy Access Utility

The Proxy utility enables managers of DECnet-RSX-11M-PLUS and DECnet-Micro/RSX systems to establish and monitor Proxy Access. Proxy Access is a feature that gives a user on one node in a network access to an account on another node without knowing either the account name or the account access control. Proxy Access is accomplished by setting up a mapping between the remote user and a local account.

2

Network Management Components

This chapter outlines DECnet-RSX network management components and refers you to related CFE, NCP, and VNP commands and parameters that you can use to configure your system.

The components described in this chapter can be modified in three different databases, depending on the network management utility command that is used, as outlined in the following list:

Command Function	CFE	NCP	VNP
Creates/modifies parameters	DEFINE	SET	SET
Clears parameters	PURGE	CLEAR	CLEAR
Displays component data	LIST	SHOW	SHOW

There is a common set of commands and parameters that apply to all DECnet systems. In addition, each DECnet system has commands and parameters that are specific to it; for example, DECnet-RSX has system-specific commands to define aliases, objects, and processes. Appendix A summarizes all CFE, NCP, and VNP commands and uses red ink to identify those that are RSX system specific. The complete command syntax for CFE, NCP, and VNP is documented in the *DECnet-RSX Guide to Network Management Utilities* .

The components that are covered in this chapter are shown in the following list with references to the sections in which they are discussed. Where applicable, descriptions include the component ID format, a table of the network management utility parameters associated with the component, and any other information that is useful in configuring the component within the network.

Component	Section
Nodes (includes access control, aliases, Ethernet addressing)	Section 2.1
Routing component (includes areas)	Section 2.2
Objects	Section 2.3
Lines	Section 2.4
Circuits	Section 2.5
Logging	Section 2.6
Counters	Section 2.7
Processes	Section 2.8
RSX–11 PSI modules	Section 2.9
X.25 protocol module	
X.25 server module	
X.29 server module	
X.25 access module	
System component	Section 2.10

2.1 Nodes

Nodes are Digital systems that run the DECnet software responsible for providing the interface between the local operating system and the network.

Three terms are applied to nodes, reflecting your relationship to the node being described:

- **Local node.** Your node—the node from which you operate.
- **Remote node.** Any node other than yours in the network.
- **Executor node.** The node at which your current network manager command executes. Usually the executor is the local node. However, you can execute most NCP commands at a remote node using the NCP TELL command. In such a case, your local node would be considered a remote node by the executor.

Many functions that the system manager performs require the identification of a specific node. For example, you can use a remote node name with the Network Display Program (NTD) to display active network information for that node. During network generation, you define your node's name and address as the executor node. You can also define node names and addresses for remote nodes. The information that you define for the executor is used in the operation of DECnet at that node.

When you configure a network at the local node, you must have node name and address entries in the volatile database for the local node and for all nodes with which you may want to establish a logical link. The following sections describe node identification and the relevant node parameters for establishing an operational network node database.

2.1.1 Node Identification

A node ID can be specified in two forms:

- **Node address.** The numeric address of the node. This address consists of an area number in the range 1 to 63 and a number in the range 1 to 1023 separated by a period. The second number represents the node number within the specified area. For example, a node address of 4.105 would represent a node in area 4 with the number of 105. If you are referencing a node within your area, you may omit the area number and the period separator. When referencing a node in a single area network, you do not have to specify the area number (1) and period. The concept of areas is explained in the discussion of routing found in Section 2.2.
- **Node name.** A unique alphanumeric character string containing 1 to 6 characters, including at least 1 alphabetic character.

To satisfy routing requirements, each node in the network must have a unique node address, which is defined at network generation. Each node must also have a unique name associated with its address in order for a logical link to be established with it. If no node name has been associated with a node address at network generation, you can do so later using a SET/DEFINE NODE command. Note that the node name is known only to the local node network software, while the node address is known network-wide by the routing function.

Once you have specified both a node name and a node address, you can use either one wherever you need to specify a node ID in CFE, NCP, and VNP commands. The local DECnet-RSX software translates node names into node addresses.

To simplify remote node identification, you can assign alias node names to destination nodes. See Section 2.1.1.2. Alias node names are known only to the local node. The network software maps alias node names to the network destination node name and address.

When you wish to execute a network management command for the executor node, you can often use the keyword EXECUTOR (EXE) as a short cut to specifying NODE *node-id*. For example, if you were executing commands on node BOSTON, the following two commands would result in the same display:

```
NCP> SHOW NODE BOSTON

NCP> SHOW EXE
```

To remove the association of a node name with a node address in the volatile database, use the CLEAR NODE NAME command. Thereafter, your node will not recognize that node name and can no longer establish logical links to that node. To remove a node from the permanent database, you must use the PURGE NODE ALL command.

The PURGE, LIST, and SHOW NODE commands allow you to specify a command for all known nodes, instead of for just one specific node. The KNOWN NODES keyword refers to all nodes that have been defined in the permanent database (and – for SHOW KNOWN NODES—in the volatile database). The SHOW NODE command also allows other more restricted plural specifications for nodes. See the discussion of SHOW commands in Section 3.3.1.3.

Loop nodes. A loop node is a special node that designates a loop path without regard for standard network routing procedures. A loop node name is associated with a specific circuit coming from the executor node. Loop nodes are used primarily for testing network hardware. For more information on loop testing, see Chapter 4.

2.1.1.1 Access Control

Certain NCP commands require you to specify access control information with the node ID when you are attempting to access a remote node. This section describes the access control format and briefly describes how to set up default access control information using alias node names.

The system manager can control user access to the local system at the node or object level for inbound logical link connections. Incoming call handling for access to local DTE and RSX-11 PSI objects is regulated differently. See the *RSX P.S.I. (Packetnet Systems Interface) RSX-11 PSI System Manager's Guide.*

Access control information can be supplied in either of two formats: the standard DECnet format or an RSX-specific format.

Standard DECnet format:

node-id [USER *user-id*]
[PASSWORD *password*]
[ACCOUNT *account*]

RSX-specific format:

node-id[/*user-id*[/*password* [/*account*]]]

where

node-id is an alphanumeric string that is the node name or address of the node to be accessed. See Section 2.1.1.

user-id is an alphanumeric string of up to 39 characters that is the user's log-in identification to be verified by the destination node. The log-in identification for RSX can be the account name or the account number. An account number can be specified in two ways: with brackets or without brackets. For example, the same account number could be expressed as [100,3] or 100,3.

password is an alphanumeric string of up to 39 characters to be verified against the destination node's system account file.

account is an alphanumeric string of up to 39 characters that supplies additional accounting information for the destination node. This field is ignored by RSX nodes.

If any string contains a blank character or tab, enclose the string with quotation marks ("*string*"). If you want to use a quotation mark within a quoted string, use double quotation marks ("") to distinguish it from a string delimiter.

NOTE

If you do not want a password echoed to your terminal as you enter an NCP command, enter a carriage return after the keyword PASSWORD. NCP prompts for the password and turns off echoing until the next time it prompts.

2.1.1.2 Alias Node Names

One way to simplify the process of specifying access control is to set up alias node names that include access control information. Alias node names are pseudonames that you can use in place of a node ID whenever you identify a DECnet node in the network. Aliases can also be used by other DECnet programs and utilities such as NFT.

You can create, modify, display, and clear alias node names in the volatile database and in the system image file by using the NCP/VNP SET ALIAS, SHOW ALIAS, and CLEAR ALIAS commands. The following example shows how to set an alias node name:

```
NCP> SET ALIAS VAX DESTINATION CHI/[1,1]/DECNET TERMINAL TT11: RET
```

This command equates the alias name VAX with node CHI. Once this command has been executed, the alias name VAX can be used in NCP commands where you would normally have to specify CHI and its access control information. The above command specifies that the alias is valid on operations initiated from terminal TT11:. Therefore, all logical link connect requests from programs associated with terminal TT11: to node VAX are transmitted to node CHI. Similarly, an NCP SHOW NODE VAX command will display data for node CHI.

Privileged users can specify aliases for a global scope—that is, for all terminals on the local node. Nonprivileged users can set aliases only for their own terminals (default). Any alias must be unique within its scope. Alias names are known only to your local node. DECnet-RSX software handles the translation from alias to node name.

DECnet-RSX automatically defines a blank alias node name for all network users. Connects to a blank node name default to the local node. You can change this association by setting a destination node for the blank alias—for example,

```
NCP> SET ALIAS "" DESTINATION CHI RET
```

The blank node name is useful for debugging network programs locally.

2.1.2 Specifying Access Control Verification

You can control inbound access to your node on two levels: the node level and the object level. You can specify that access control be verified for your node on all incoming connect requests to DECnet objects. In fact, node level access control verification is required on systems that support multiuser protection. To assure access control verification, you must set VERIFICATION parameters to ON at both the executor and the object level.

You can use CFE, NCP, and VNP commands to modify access control on both the node and the object level. If you set the VERIFICATION parameter to OFF in a SET/DEFINE EXECUTOR command, the executor overrides the verification options set for individual objects and allows access to all objects. When verification at the executor level is set to ON, verification states specified for individual objects are used. See the CFE/NCP DEFINE/SET OBJECT command descriptions in the *DECnet-RSX Guide to Network Management Utilities* .

CAUTION

If access control verification is turned off, anyone can access your system.

2.1.3 Proxy Access

Proxy Access is a feature that enables a user on one node in a network to access an account on another node. With Proxy Access, authorized users do not need to specify (or even know) the name of the destination node or any access control information. Additionally, system managers use INCOMING PROXY and OUTGOING PROXY parameters of the SET and DEFINE EXECUTOR commands to control access to and from their local nodes.

2.1.4 Node Parameters

The permanent database must contain certain information about the local node. Optionally, it can contain information for all nodes with which you wish to communicate. You can specify names and node addresses for any or all of the remote nodes. If a remote node can be down-line loaded, you can specify a number of default parameters to be used locally to perform a down-line load operation. Table 2–1 lists all node parameters by function and groups them according to the kind of node to which they apply.

You can modify and display node parameters by using the following commands which are described in the DECnet-RSX Guide to Network Management Utilities:

CFE Commands	NCP/VNP Commands
DEFINE EXECUTOR	SET EXECUTOR
DEFINE NODE	SET NODE
———	CLEAR EXECUTOR
PURGE NODE	CLEAR NODE
LIST EXECUTOR	SHOW EXECUTOR
LIST NODE	SHOW NODE

Table 2–1: Node Parameters and Their Functions

Executor Node	Remote Node
Parameter Function—Node Identification	
ADDRESS	ADDRESS
NAME	NAME
IDENTIFICATION	
Parameter Function—Local Node State	
STATE (ON/OFF/SHUT)	
Parameter Function—Access Control	
VERIFICATION STATE (OFF/ON)	
INCOMING PROXY[1]	
OUTGOING PROXY[1]	
Parameter Function—Logical Link Control	
MAXIMUM LINKS	
MAXIMUM NODE COUNTERS	
SEGMENT BUFFER SIZE[2]	
Parameter Function—Routing Control[3]	
MAXIMUM ADDRESS	
MAXIMUM AREAS	

[1]DECnet–11M–PLUS only.

[2]See Chapter 6 for more information on specifying system buffers.

[3]See Section 2.2 for a discussion of routing and routing parameters.

Table 2–1 (Cont.): Node Parameters and Their Functions

Executor Node	Remote Node
Parameter Function—Routing Control[3]	
MAXIMUM COST	
MAXIMUM HOPS	
AREA MAXIMUM COST	
AREA MAXIMUM HOPS	
MAXIMUM BROADCAST ENDNODES	
ROUTING TIMER	
BROADCAST ROUTING TIMER	
MAXIMUM PATH SPLITS[1]	
Parameter Function—Routing Initialization Passwords[3]	
TRANSMIT PASSWORD	
RECEIVE PASSWORD	
Parameter Function—Loop Node Identification	
	CIRCUIT
Parameter Function—Down-Line Loading[4]	
	DIAGNOSTIC FILE
	HARDWARE ADDRESS
	HOST

[1]DECnet–11M–PLUS only.

[3]See Section 2.2 for a discussion of routing and routing parameters.

[4]See Chapter 5 for information on down-line loading and up-line dumping.

Table 2–1 (Cont.): Node Parameters and Their Functions

Executor Node	Remote Node
Parameter Function—Down-Line Loading[4]	
	LOAD FILE
	SECONDARY LOADER
	SERVICE CIRCUIT
	SERVICE DEVICE
	SERVICE NODE VERSION (PHASE III/PHASE IV)
	SERVICE PASSWORD
	TERTIARY LOADER
Parameter Function—Up-Line Loading[4]	
	DUMP ADDRESS
	DUMP COUNT
	DUMP FILE
	HARDWARE ADDRESS
Parameter Function—Incoming RSX–11 PSI Call Control	
SUBADDRESS *range*	

[4]See Chapter 5 for information on down-line loading and up-line dumping.

2.1.4.1 Executor Node Identification String

During network generation, you can provide a string of information that is displayed whenever you display executor node information. To modify this string, use the IDENTIFICATION parameter with the DEFINE EXECUTOR command. If the string you specify contains space or tab characters, you must delimit the string with quotation marks. If you want to include a quoted string within a displayed string, use double quotation marks around the quoted string to distinguish it from the string's end delimiter. When the message is displayed, the system will discard one set of quotation marks.

Example:

```
CFE> DEFINE EXECUTOR IDENTIFICATION "DECnet RSX""BOS""V4.1"[RET]
```

2.1.4.2 Executor Node Subaddresses

During network generation, you can define a range of subaddresses for the executor node to use for DLM operations. The executor node accepts incoming DLM calls that have a subaddress that falls within the specified range. An alternative method for specifying which incoming calls will be accepted for DLM circuits is to use the circuit NUMBER parameter as described in Section 2.5.5.1. This method must be used when the network being used does not support subaddresses.

Use the SUBADDRESSES parameter to modify the executor subaddresses in any of the three databases. A subaddress can be a single number or a range of numbers in the range of 0 to 9999. When specifying a range, separate the two subaddresses that delimit the range with a hyphen, always listing the smaller number first.

Example:

```
NCP> SET EXECUTOR SUBADDRESSES 42-50 [RET]
```

This command specifies the executor node to handle all incoming RSX–11 PSI calls that specify a local DTE subaddress in the range of 42 to 50. All other calls will be handled by RSX–11 PSI.

2.1.5 Node Counters

DECnet software automatically collects certain statistics, called counters, for nodes in the network. Such information can include the number of connects sent and received, the number of messages sent and received, and the number of messages retransmitted.

If a logical link is established to a new node after all node counter blocks have been used, the least recently used counter block is reassigned to the new node, and the counters for the old node are displayed in an event message.

For more information on using counters, see Section 2.7. For a complete listing of node counters, see Appendix E.

2.1.6 Ethernet Address of Node

Nodes on Ethernet lines are identified by unique Ethernet addresses. A message can be sent to one, several, or all nodes on an Ethernet line simultaneously, depending on the type of Ethernet address used. You do not normally have to specify the Ethernet address of an individual node in order to configure your network; the software at the node sets its own Ethernet address. You need to know the Ethernet address of a node for service functions, such as down-line loads or circuit loopback tests, but not for normal network operations.

2.1.6.1 Format of Ethernet Addresses

Ethernet addresses are represented as six pairs of hexadecimal digits separated by hyphens, for example, AA-00-03-00-67-FF.

Xerox Corporation assigns a block of addresses to a producer of Ethernet interfaces upon application. Thus every manufacturer has a unique set of addresses to use. Normally, one address out of the assigned block of physical addresses is permanently associated with each interface and is usually in a read-only memory. This address is known as the **Ethernet hardware address** of the interface.

NOTE

Use the NCP SHOW LINE *line-id* CHARACTERISTICS command to display the hardware address. See the example in Chapter 3.

Digital's interface to Ethernet is one of the following communication controllers: the DEUNA, DEQNA, DELUA, or the DELQA. At the node, the controller has the ability to set a different logical address to be used by the interface: this address is known as the **Ethernet physical address.** When a node on the Ethernet initially starts up, the physical address is the same as the Ethernet hardware address. Then when DECnet turns on a UNA or QNA device, DECnet constructs a physical address by appending the local node's node address to a constant 8-digit number derived from the block addresses assigned to Digital (AA-00-04-00).

Once the Ethernet physical address has been set to its new value, it is reset to its original hardware address value only when a reset is issued to the communication controller (DEUNA, DEQNA, DELUA, or DELQA), such as when the machine power is shut off.

2.1.6.2 Ethernet Multicast Address Types

Ethernet physical addresses, described in the preceding section, and Ethernet multicast addresses, described in the following section, are distinguished by the value of the leading low-order bit of the first byte of the address:

- **Physical address.** The unique address of a single node on any Ethernet (low-order bit = 0).
- **Multicast address.** A multidestination address of one or more nodes on a given Ethernet (low-order bit = 1).

There are two types of multicast address:

- A **multicast group address** is an address that is assigned to any number of node groups so that they are all able to receive the same message in a single transmission by a sending node.
- A **broadcast address** is a single multicast group address (specifically, FF-FF-FF-FF-FF-FF) to which a message can be sent if it must be received by all nodes on a given Ethernet. Use a broadcast address only for messages to be acted upon by all nodes on the Ethernet, since all nodes must process them.

2.1.6.3 Ethernet Physical and Multicast Address Values

Digital physical addresses are in the range AA-00-00-00-00-00 through AA-00-04-FF-FF-FF. Multicast group addresses assigned for use in cross-company communications are:

Value	Meaning
FF-FF-FF-FF-FF-FF	Broadcast
CF-00-00-00-00-00	Loopback assistance

Digital multicast group addresses assigned to be received by other Digital nodes on the same Ethernet are:

Value	Meaning
AB-00-00-01-00-00	Dump/load assistance
AB-00-00-02-00-00	Remote console
AB-00-00-03-00-00	All Phase IV routers
AB-00-00-04-00-00	All Phase IV end nodes
AB-00-00-05-00-00 through AB-00-04-FF-FF-FF	Reserved for future use

DECnet always sets up the communication controller (the DEUNA, DEQNA, DELUA, or DELQA) at each node to receive messages sent to any address in the above list of Digital multicast addresses.

2.2 Routing Component

Routing is the network function that determines the path or route along which data, or packets, travel from their source to their destination. Routing is performed transparently by the Routing layer of DECnet software. As network manager, however, you must be concerned with the configuration of the network into areas, routing and nonrouting nodes, and with a set of parameters that provides a degree of indirect control over network routing. This section explains the concept of areas, the different types of routing and nonrouting nodes, and describes the routing parameters you can control.

2.2.1 Areas

DECnet-RSX Phase IV supports the concept of network areas. Network areas allow a larger network than was previously possible and also allow a large network to be split up into smaller units called areas. Using areas you can establish a hierarchical network in which there is a network of areas with each area made up of a network of nodes. This two-level approach must be carefully planned at the time the network is designed. In general, you should set up areas to correlate roughly with the nature of your network traffic. You should have high traffic volumes within the area and lower traffic volumes across area boundaries. Chapter 7 of this manual and the *DECnet-RSX Network Generation and Installation Guide* contain detailed area configuration guidelines that must be followed when setting up areas in a network.

During network generation you must specify which area your node will be in. There are also several routing parameters that are specific to nodes that perform area routing. These area routing parameters are discussed in the Section 2.2.3.

2.2.2 Types of Nodes

Phase IV DECnet allows communication with five types of nodes:

- Phase IV level 2 routing nodes
- Phase IV level 1 routing nodes
- Phase IV nonrouting nodes (end nodes)
- Phase III routing nodes
- Phase III nonrouting nodes (end nodes)

Phase IV level 2 routing nodes have a route-through capability and are able to support multiple circuits. They maintain one routing database that allows them to route their own transmit packets as well as packets received from other nodes within their own area. In addition, they maintain a second routing database for the areas within the network. This database allows them to forward inter-area packets to level 2 routing nodes that are outside their local area. Phase IV level 2 routing nodes are sometimes referred to as area routing nodes.

Phase IV level 1 routing nodes, like Phase IV level 2 routing nodes, have a route-through capability and are able to support multiple circuits. They maintain a single routing database that allows them to route their own transmit packets as well as packets received from other nodes within their own area. Phase IV level 1 routing nodes cannot forward packets to nodes outside their area directly; however, they can route packets to areas outside their own by sending them to the nearest Phase IV level 2 routing node. On an Ethernet, a single routing node called the designated router performs routing functions. See the Ethernet circuit parameters description in Section 2.2.3.11. for all of the nodes on the Ethernet. In area networks, this node may be a Phase IV level 2 routing node.

Phase IV end nodes support only one active circuit and do not maintain a routing database. An end node can establish logical links to any node in the network, but it can send packets to and receive packets from the adjacent node only. Because all nodes on an Ethernet are considered to be adjacent to each other, an Ethernet end node can communicate directly with more than one node. End nodes connected by a DDCMP, PCL, or DLM circuit, however, are adjacent to a single node only.

Phase III routing nodes adhere to the DNA Phase III architecture. Although DECnet-RSX no longer supports generation of these nodes, it does support communication with an existing Phase III routing node. These nodes have most of the capabilities of a Phase IV level 1 routing node but cannot send packets to nodes outside their area or to nodes with node numbers above the Phase III address limit of 255. There are additional configuration issues to be considered when a mixed Phase III/Phase IV network is present. For more information on these types of networks see the configuration guidelines in the *DECnet-RSX Network Generation and Installation Guide* and Chapter 7.

Phase III end nodes also adhere to the DNA Phase III architecture. Although DECnet-RSX no longer supports generation of these nodes, it does support communication with an existing Phase III end node. Phase III end nodes cannot send packets to nodes outside their area or to nodes with node numbers above the Phase III address limit of 255. In addition, Ethernet circuits are not supported on Phase III end nodes.

2.2.3 Routing Parameters

Some parameters that affect routing are defined for the executor node component, while others are defined for the circuit component. You can modify and display routing parameters by using the following commands which are described in the *DECnet-RSX Guide to Network Management Utilities* :

CFE Commands	NCP/VNP Commands
DEFINE EXECUTOR	SET EXECUTOR
———	CLEAR EXECUTOR
LIST EXECUTOR	SHOW EXECUTOR
DEFINE CIRCUIT	SET CIRCUIT
PURGE CIRCUIT	CLEAR CIRCUIT
LIST CIRCUIT	SHOW CIRCUIT

Table 2–2 lists the parameters by component and specifies the utilities that can modify them. The network generation procedure sets default values for all of these parameters. Parameters that apply to Ethernet nodes only are listed separately.

Table 2–2: Routing Parameters

Component	Parameter	CFE	NCP/VNP
General routing parameters:			
EXECUTOR	MAXIMUM ADDRESS	X	
	MAXIMUM AREAS	X	
	MAXIMUM COST	X	
	MAXIMUM HOPS	X	
	AREA MAXIMUM COST	X	
	AREA MAXIMUM HOPS	X	
	ROUTING TIMER	X	X
	MAXIMUM PATH SPLITS[1]	X	
CIRCUIT	COST[2]	X	X
	LEVEL 1 COST[1]	X	X
	LEVEL 2 COST[1]	X	X
	HELLO TIMER	X	X
Ethernet-only parameters:			
EXECUTOR	BROADCAST ROUTING TIMER	X	
	MAXIMUM BROADCAST ENDNODES	X	
CIRCUIT	MAXIMUM ROUTERS	X	
	ROUTER PRIORITY	X	

[1]DECnet–11M–PLUS only.

[2]DECnet–11M/S systems only.

2.2.3.1 MAXIMUM ADDRESS Parameter

During network generation of a routing node, you specify the highest node address to be allowed in the network. This controls the size of routing configuration messages exchanged between nodes and determines the size of the internal routing database constructed by the network software. You can modify this value in the permanent database using the MAXIMUM ADDRESS parameter. To obtain the most efficient routing operation, sequentially number your nodes from 1 to *n*, where *n* is the highest node address.

2.2.3.2 MAXIMUM AREAS Parameter

If the node is a level 2 routing node, you must specify the highest area number to be allowed in the network. This network generation parameter controls the size of the area routing database and the area routing configuration messages exchanged between level 2 routing nodes. You can modify this value in the permanent database using the MAXIMUM AREAS parameter. To obtain the most efficient area routing operation, sequential area IDs should be used.

2.2.3.3 MAXIMUM COST and MAXIMUM HOPS Parameters

You can alter the values set for the MAXIMUM COST and MAXIMUM HOPS parameters, which determine the reachability of nodes in the network.

The MAXIMUM COST parameter specifies the maximum total path cost allowed from the executor node to any other network node. A remote node is unreachable if the cost to get to the remote node exceeds the value set for this parameter. Use as small a number as possible in the range of 1 to 1022.

The MAXIMUM HOPS parameter specifies the maximum number of hops allowed from the executor to any other node in the network. The largest value for this parameter in a DECnet network defines the DECnet network diameter. A remote node is unreachable if the number of hops required to reach it exceeds the value set for this parameter. Use as small a number as possible in the range of 1 to 30.

2.2.3.4 AREA MAXIMUM COST and AREA MAXIMUM HOPS Parameters

These two parameters determine the reachability of areas in the network in the same manner that the MAXIMUM COST and MAXIMUM HOPS parameters determine reachability of nodes within an area.

The AREA MAXIMUM COST parameter specifies the maximum total path cost allowed from the local area to any other area in the network. A remote area is considered unreachable if the cost to that area exceeds the value set for this parameter. Use as small a number as possible in the range of 1 to 1022.

The AREA MAXIMUM HOPS parameter specifies the maximum number of hops allowed from the local area to any other area in the network. A remote area is unreachable if the number of hops to that area exceeds the value set for this parameter. Use as small a number as possible in the range of 1 to 30.

2.2.3.5 ROUTING TIMER and BROADCAST ROUTING TIMER Parameters

DECnet-RSX transmits a routing configuration message to all adjacent nodes whenever a circuit, line, and node configuration change occurs. Routing configuration messages provide dynamic network configuration information that can affect data packet routing. For example, if a network user changes the state of a circuit and thereby renders a remote node unreachable, this change is reflected automatically in the routing configuration data transmitted to adjacent nodes. If no changes occur for an extended period of time, these configuration messages are transmitted as a result of a timer expiring. This timer is specified using the ROUTING TIMER or BROADCAST ROUTING TIMER parameter.

On non-Ethernet nodes, you use the ROUTING TIMER parameter to set the timer whose expiration forces a routing update regardless of whether the network configuration has altered. Use a timer value in the range of 1 to 65535.

For a node on an Ethernet circuit, the timing mechanism is called a broadcast routing timer. When the timer expires, the local node sends a multicast routing configuration message to all nodes on the Ethernet. All nodes are considered to be adjacent. You can use the BROADCAST ROUTING TIMER parameter to specify the frequency at which this message is transmitted. The range is: 1 to 65535. This timer is usually set to a value approximately 30 to 40 seconds lower than the routing timer for a node on a non-Ethernet circuit. The routing timer on a non-Ethernet node is set for every few minutes. Ethernet routing messages are sent more frequently so that full routing messages can be exchanged in case of datagram loss.

2.2.3.6 MAXIMUM PATH SPLITS Parameter (DECnet–11M–PLUS only)

On DECnet–11M–PLUS systems, when you send traffic to a destination node, using multiple circuits can improve performance. This parameter enables you to specify the number of equal cost paths to split between. This feature is only supported on networks that have all RSX nodes or other type nodes that support out-of-order segment caching.

2.2.3.7 COST Parameter (DECnet–11M and -11S Only)

During network generation, a cost is automatically assigned to each circuit connecting the local node to an adjacent node. DECnet software uses circuit cost values to determine the path over which data is transmitted. You can use the COST parameter to change the cost of a circuit. The range is: 1 to 25. Altering circuit costs can change packet-routing paths and thereby affect the use and availability of network circuits and resources.

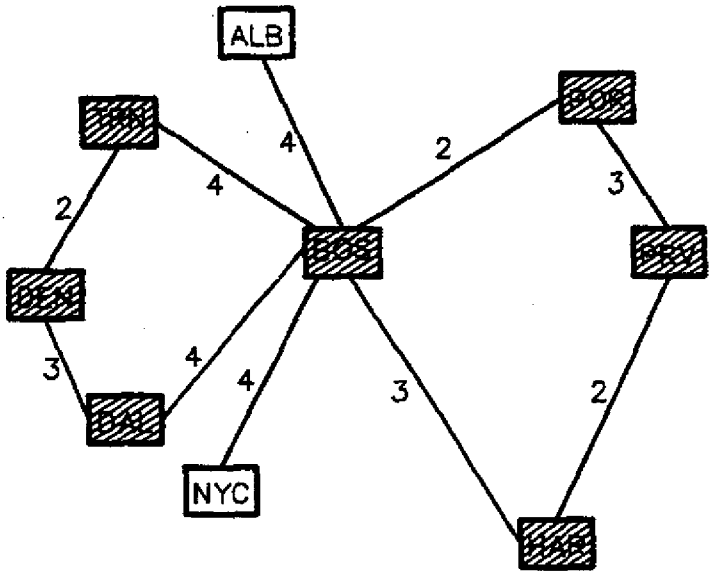
When a network installation is performed the following guidelines are automatically used when first assigning circuit costs; you can adjust these values later if doing so will improve performance.

Ethernet circuits	3
Line speeds over 9600	5
Line speeds of 9600 or less	7
DLM circuits	15

DECnet–11M–PLUS systems have an added feature called **equal cost load splitting**. Equal cost load splitting improves message transmission time by checking the queue backup of all paths to the destination that have the same cost. The message is then transmitted over the least congested equal-cost path. For example, in Figure 2-1, there are two equal-cost paths from BOS to PRV (paths #3 and #4). With equal cost load splitting, the system compares the queue backup of the two paths and uses the path with the least backup.

Figure 2-1 illustrates the relationship between circuit and path costs.

Figure 2–1: Circuit and Node Path Cost Relationship



LEGEND	PATH	LINE COST	PATH COST	HOPS
= ROUTING NODE	#1	TRN DEN DAL (2, 3)	5	2
= NON–ROUTING NODE	#2	TRN BOS DAL (4, 4)	8	2
	#3	BOS POR PRV (2, 3)	5	2
	#4	BOS HAR PRV (3, 2)	5	2

LKG–1174–87

2.2.3.8 LEVEL 1 COST and LEVEL 2 COST Parameters (DECnet-11M-PLUS Only)

These two parameters specify the routing cost of a circuit. Both level 1 and level 2 routing attempt to route messages along the path with the lowest cost.

LEVEL 1 COST pertains to level 1 routing only and specifies the cost of a circuit when routing messages between two nodes in the same area. LEVEL 2 COST pertains to area routers only and specifies the cost of a circuit when routing messages between two areas. The acceptable range for both parameters is 1 to 25.

2.2.3.9 HELLO TIMER Parameter

The DECnet Routing layer software sends hello messages over DECnet circuits at regular intervals and waits for a response to determine if the circuit is still operative. If the circuit is inactive, the Routing layer attempts to reinitialize it. You can use the HELLO TIMER parameter to modify the frequency with which hello messages are transmitted. The range is: 1 to 8191.

2.2.3.10 MAXIMUM BROADCAST ENDNODES Parameter (Ethernet Nodes Only)

This parameter controls the number of end nodes you can communicate with directly without going through an intermediate routing node. You can modify the value in the permanent database by using the MAXIMUM BROADCAST ENDNODES parameter. When you modify this value keep in mind that this value is the total of all broadcast circuits on the node. Use a value as small as possible in the range 1 to 65,534.

2.2.3.11 Ethernet Circuit Parameters

There are two routing parameters that can be specified for Ethernet circuits in the DEFINE CIRCUIT command:

ROUTER PRIORITY—When two or more routing nodes are connected to an Ethernet cable, the routing node with the highest priority (range of 0 to 127) becomes the router designated to provide routing services for the end nodes on the Ethernet. End nodes on the same Ethernet cable can communicate directly with one another; routers are required to send messages to nodes that are not connected to the Ethernet. If two nodes share the highest priority, the node with the highest node address is selected. To learn which router is the designated router, issue a SHOW CIRCUIT command.

Example:

```
NCP> SHOW CIRCUIT UNA-0 CHARACTERISTICS RET

Circuit characteristics as of 3-OCT-86 15:03:27

Circuit = UNA-0

   Service = Enabled
   Adjacent node = 4.36(SAM), Designated router = 4.36(SAM), Cost = 3
   Hello timer = 15, Listen timer = 66
   Owner = XPT
   Type = Ethernet
```

MAXIMUM ROUTERS – This parameter specifies the maximum number of routers other than the executor node, to be allowed on the identified Ethernet circuit. The practical maximum number of routers is about 10 because of the control traffic overhead which is composed of routing messages and hello messages. The absolute maximum number of routers is 32.

2.3 Objects

An object is a DECnet Application layer task or module that can communicate with another such task or module over a logical link.

2.3.1 Object Types

All objects are identified by an object type code of either 0 or an integer in the range of 1 to 255, depending on the kind of object.

- **Named objects** are installed tasks identified by any valid user-defined RSX task name in a logical link connect request. All named objects have an object type code of 0.
- **Numbered objects** are installed tasks or modules such as FAL and NICE, that provide a generic, cross-system, network service. Each numbered object is identified by an object type code in the range of 1 to 255. Each type code always represents the same function within a network, even if the task that actually performs the function has a different name from node to node. Numbers in the range of 1 to 127 are reserved for Digital-supplied services; numbers from 128 through 255 are available for customer-supplied services. Appendix B lists all DECnet-RSX object type codes.

2.3.2 Object Parameters

At network generation, the system automatically defines objects for all selected tasks. Later, you can use the DEFINE/SET commands to create or modify a task or object, using your own value or an object type code from Appendix B to specify the object. You can modify and display object parameters by using the following commands which are described in the DECnet-RSX Guide to Network Management Utilities:

CFE Commands	NCP/VNP Commands
DEFINE OBJECT	SET OBJECT
PURGE OBJECT	CLEAR OBJECT
LIST OBJECT	SHOW OBJECT

Parameters that you can specify for each type of object are summarized in Table 2–3.

Table 2–3: Object Types and Parameter Functions

Object Types and Parameter Functions	Parameter
Named and Numbered Objects	
Specifies the UIC the object is to run under	USER
Specifies the state of access control verification	VERIFICATION
Numbered Objects Only	
Specifies the number of instances of an object allowed	COPIES
Specifies the task name of an object	NAME

Since all named objects have the same type code (0), any changes that you make to object type 0 apply to all named objects on your node. For programming information on named objects and their use in task-to-task communication, refer to the DECnet-RSX Programmer's Reference Manual and the RSX–11 PSI User's Guide.

2.3.2.1 Object UICs and Access Verification

Once activated, objects run under a particular UIC. You can use the USER parameter to specify whether the object runs under the log-in UIC or under the default UIC under which it was built or installed. The log-in UIC is the user ID that was specified in the access control information provided at the time of the connection. If a log-in UIC is not provided, the object runs under the default UIC.

The object's access control verification option must be set to ON or INSPECT to run under log-in UICs. If the object's access control verification option is not on, the object runs under the default UIC. See Section 2.1.2 for a discussion of the VERIFICATION parameter and access control verification.

2.3.2.2 Single Copy and Multicopy Objects

Use the COPIES parameter to specify the number of copies of a task that can be run at the local node at once. Specify SINGLE for one copy or specify a number in the range of 2 to 64 for a multicopy object. All numbered object types can have multiple copies.

When you specify a number, each incoming request is sent to a new copy of the task. Thus, each copy of the task handles one incoming connection. This is referred to as single-threaded operation.

When you specify SINGLE, the task itself determines the number of simultaneous connections it can have. This is referred to as multi-threaded operation.

2.3.2.3 Object Names

When you install a task as a numbered object, you can use the NAME parameter to specify any task name that is valid for RSX. If a numbered object has multiple copies, then you must install it with a name in the following format: *tsk*$$$, where *tsk* is any valid 3-character RSX task name. If the task name is not installed in this manner, a logical link connection cannot be established.

When the network software establishes the logical link, it replaces the three dollar signs with a specific number. You can see the final task names in the NTD display of active tasks. See the NTD chapter in the *DECnet-RSX Guide to Network Management Utilities* .

2.4 Lines

Lines, and their associated device driver processes, provide the low-level communication functions that are used to provide the circuits over which DECnet communication takes place. See Section 2.5. DECnet-RSX supports four types of lines: DDCMP, PCL, Ethernet, and RSX–11 PSI. A DDCMP line provides the physical point-to-point or multipoint connection between two or more nodes, PCL and Ethernet lines are multiaccess connections between two or more nodes, and an RSX–11 PSI line is the physical link between a user's data terminal equipment (DTE) and a Packet Switching Data Network (PSDN).

Just as you must establish node parameters, you must also establish parameters for all physical lines connected to the local node. Thus you must identify each line by name and specify information that directly affects the line's operation.

2.4.1 Lines and Line Devices

Table 2–4 lists the line devices that provide the four types of lines supported by DECnet-RSX software: DDCMP, PCL, Ethernet, and RSX–11 PSI. All line devices are referenced using their 2- or 3-letter device driver name shown in the table. The table also shows the multiline and multipoint features available with each line device. Each line type is described in the following sections. You should be familiar with the entire range of line devices and their impact on network management.

Table 2–4: Digital Communication Devices Supported by DECnet-RSX

Line	Device Name	Multiline	Multipoint Device
DDCMP Lines			
DHU11	DHU	Yes	Yes
DHV11/DHQ11	DHV	Yes	Yes
DL11	DL	No	Yes
DLVE1/DLVJ1	DLV	No	Yes
DMC11/DMR11	DMC	No	No
DMP11	DMP	No	Yes
DMV11	DMV	No	Yes
DPV11	DPV	No	Yes
DU11	DU	No	Yes
DUP11	DUP	No	Yes
DUV11	DUV	No	Yes
DV11	DV	Yes	Yes
DZ11	DZ	Yes	Yes
DZV11	DZV	Yes	Yes
KMC11+DUP11	KDP	Yes	Yes
KMC11+DZ11	KDZ	Yes	Yes
Ethernet Lines			
DEUNA/DELUA	UNA	N/A	N/A
DEQNA/DELQA	QNA	N/A	N/A

Table 2–4 (Cont.): Digital Communication Devices Supported by DECnet-RSX

Line	Device Name	Multiline	Multipoint Device
RSX–11 PSI Lines			
KMS11-BD/BE	KMX	Yes	N/A
KMS11-PX/PY	KMY	No	N/A
KMV11-A	KMV	Yes	N/A
DUP11	SDP	No	N/A
DPV11	SDV	No	N/A
PCL Lines			
PCL11-B	PCL	N/A	N/A

2.4.1.1 DDCMP Line Devices

Four of the DDCMP line devices listed in Table 2–4 have the DDCMP microcode contained in the hardware device: DMC-11, DMR-11, DMP-11, and DMV-11. The other DDCMP devices in Table 2–4 have a software DDCMP process.

The DMC-11 and the DMR-11, which operate identically, provide a point-to-point connection between two nodes.

The DMP-11 and the DMV-11 can be point-to-point, multipoint control, or multipoint tributary line devices. Therefore, they can provide a multipoint connection between more than two nodes. See Section 2.5.4.1 for more information about multipoint operation. It is possible to connect two multipoint lines to the same node. The node can then serve as the control station for one multipoint line while acting as a tributary on another multipoint line.

2.4.1.2 PCL Line Devices

The PCL-11B provides point-to-point connections by means of a parallel time division multiplexed bus (TDM). PCL-11B hardware allocates a fraction of available bus bandwidth to each station. Each node on the PCL bus is known by a unique station address.

2.4.1.3 Ethernet Line Devices

The Ethernet line device is a DEUNA, DEQNA, DELUA, or DELQA, each of which provides a physical connection between two or more nodes on the same Ethernet cable. The Ethernet line protocol supports up to 1023 nodes on the same line.

A node is connected to the Ethernet line by a DEUNA, DEQNA, DELUA, or DELQA communication controller, a transceiver, and a transceiver cable. A specific node on the Ethernet is identified by the address of its line device. See Section 2.1.6.

2.4.1.4 RSX–11 PSI Line Devices

RSX–11 PSI supports five line devices – the DUP11-DA, the DPV11, the KMS11-BD, the KMS11-PX, and the KMV11-A. The DUP11-DA (UNIBUS) and the DPV11 (Q-bus) are low speed synchronous interfaces. The combination of the KMS11-BD controller hardware and the X.25 level 2 microcode provides a medium speed synchronous interface called the KMX. The KMS11-BD supports eight lines, only two of which can be active simultaneously. Similarly, the combination of the KMS11-PX controller hardware and the X.25 level 2 microcode provides a medium speed, single line, synchronous interface called the KMY. The KMV11-A controller hardware combined with the X.25 level 2 microcode provides a medium speed, single line, synchronous interface called the KMV.

2.4.2 Line Identification

You can use network management commands to modify parameters for an individual line or for all known lines. Use the KNOWN LINES keyword to specify all lines identified to the system during network generation. To specify an individual line, use a line ID of the form

dev-c[*-u*]

where

dev is a device name from Table 2–4.

c is a decimal number, 0 or a positive integer, designating the device's hardware controller.

u is a decimal unit number, 0 or a positive integer, included if the device is a multiple line controller.

You can use a wildcard character (*) in place of the controller or unit number to specify all lines in that category. You cannot specify a particular unit if you specify a wildcard character for the controller.

Tributaries for a multipoint line device are defined at the circuit level. See Section 2.5.2.1.

NOTE

To create a new line ID in the network databases, you must perform another network generation.

2.4.3 Line Parameters

The permanent database should contain line parameters for all physical lines connected to the local node or DTE. These parameters supply information used to control various aspects of a line's operation. You can modify and display line parameters by using the following commands which are described in the *DECnet-RSX Guide to Network Management Utilities* :

CFE Commands	NCP/VNP Commands
DEFINE LINE	SET LINE
PURGE LINE	CLEAR LINE
LIST LINE	SHOW LINE

NCP/VNP line parameters fall into two categories: loaded options and loading options. Loaded options are valid only for lines that have already been loaded into the volatile database. Loading options apply to lines that are currently being loaded.

Table 2–5 categorizes all line parameters according to line type and parameter function.

Table 2–5: Line Types and Parameter Functions

Parameters	Line Types and Parameter Functions
	All lines
CONTROLLER CSR	Specifies device hardware characteristics
UNIT CSR	
VECTOR	
PRIORITY	
SPEED	
COUNTER TIMER	Sets counter timer for event logging
STATE (CLEARED/OFF/ON)	Sets line's operational state; valid for all lines on CFE commands, for RSX–11 PSI only on NCP/VNP
DUPLEX (FULL/HALF)	Sets hardware transmission mode
CONTROLLER (LOOPBACK/NORMAL)	Establishes line level loopback control for device; applies to DMC, DMR, DMP, DMV, UNA, QNA lines only
LOCATION (FIRSTFIT/TOPDOWN)	Specifies where to load software
	Multipoint DDCMP lines
DEAD TIMER	Establishes DDCMP line-polling protocol. See the parameter descriptions in Section 2.5.4.
DELAY TIMER	
MULTIPOINT DEAD	
	RSX–11 PSI lines
HOLDBACK TIMER	Establishes frame control
MAXIMUM DATA	
MAXIMUM WINDOW	
RETRANSMIT TIMER	Controls retransmission of frames
MAXIMUM RETRANSMITS	
OWNER (DLX/PLI)	Changes line's owner

2.4.3.1 Hardware Device Parameters

During network generation, you specified a number of parameters that directly affect the operation of the hardware device for a line. You can modify this information as necessary in any of the three databases using network management commands.

The CONTROLLER CSR parameter specifies the control status register (CSR) address of the line device. If you change the controller CSR address of a multiline device, be sure to make the same change for any other lines sharing that controller.

On KMC-11 devices (KDP and KDZ), use the UNIT CSR parameter to modify the address of the first CSR of the secondary device (DUP or DZ) that the KMC is controlling. In addition, you can use the UNIT CSR parameter to modify the CSR for the KMX modem controller.

Use the VECTOR parameter to modify the address of the vector that contains the address of the interrupt service routines for a line.

Use the PRIORITY parameter to modify the hardware priority of the line device.

Use the SPEED parameter to modify the speed of the line device. The speed of a synchronous device is determined by the speed of the modem. The speed of an asynchronous device is determined by switch settings on the line device. The line speed specified in NETGEN and CFE commands is used to calculate timing criteria for the line and buffer requirements. If the specified speed is incorrect, the timing will be incorrect and the large buffer allocation may be incorrect. See Chapter 6. If the specified line speed is higher than the actual line speed, there may be CRC errors on large data blocks or large data block allocation failures. If the specified line speed is lower than the actual line speed, there is a long delay on line timeouts. For these reasons, be sure to use CFE to change the line speed on synchronous lines if the modem speed is changed.

2.4.3.2 Line States and Loading

DECnet users indirectly set line states by setting the states of DECnet circuits. See Section 2.5.3.1. RSX–11 PSI users directly set line states of RSX–11 PSI lines by using the DEFINE/SET LINE commands.

You can use the SHOW LINE command explained in Section 3.3.1. to display the state of any or all lines. In some cases, a substate is also displayed. Circuit/line states and substates are described in Section 2.5.3.1 and are summarized in Table 2–7. A complete list of RSX–11 PSI line, circuit, and module states and state transition reasons is given in Appendix G.

Line loading is directly related to the line states specified either in the permanent database or during network generation. Lines with a CLEARED state are not loaded when you load the DECnet-RSX system. Lines with an OFF state are loaded, but must be set to the ON state before being used.

The LOCATION parameter specifies the loading strategy to use when loading the software associated with the line. If the line is loaded TOPDOWN, the software is loaded at the top of the partition. If FIRSTFIT is used, the software is loaded in the first available memory location large enough to accommodate the software.

2.4.3.3 RSX–11 PSI Parameters

The following considerations should be reviewed when modifying the RSX–11 PSI-related parameters.

Frame control. In the permanent database, the MAXIMUM DATA and MAXIMUM WINDOW parameters were set to defaults for the PSDN to which the line's DTE is connected. Consult the documentation for your specific RSX–11 PSI network for details of these values before you modify them. The MAXIMUM DATA parameter specifies the maximum size of a frame and should always specify a size at least five bytes larger than the packet size specified in the protocol module for SVCs and the circuit component for PVCs. The MAXIMUM WINDOW parameter specifies the maximum number of unacknowledged frames allowed for this line.

Frame retransmission. Use the RETRANSMIT TIMER parameter to specify how long the software will wait for an acknowledgment of a frame before retransmitting the frame. Use the MAXIMUM RETRANSMITS parameter to specify the maximum number of times the frame can be retransmitted. The value specified for the timer should take into account the line speed and the frame size.

Holding acknowledgments. The HOLDBACK TIMER parameter is used to specify the length of time that an acknowledgment may be held back in order to be included with another data message.

Line ownership. Use the OWNER parameter to specify the ownership of an RSX-11 PSI line. The only permitted owners are the RSX-11 PSI software (PLI) or the line service software (DLX).

RSX-11 PSI line counters. Use the COUNTER TIMER parameter to control the frequency with which the line counters are logged and zeroed. For more information on line counters see Section 2.4.4. To stop line counters from being logged, use the CLEAR LINE command.

2.4.4 Line Counters

DECnet software automatically maintains statistics, called counters, for most lines in the network. Line counters for DDCMP lines include a count of the local and remote process errors, and the amount of time since the counters were last zeroed. DECnet-RSX maintains these counters for all DDCMP lines except DMC/DMR and DMP/DMV point-to-point lines.

Line counters for Ethernet lines include the number of bytes, multicast bytes, data blocks, and multicast blocks sent and received; the number of blocks deferred or sent after collision; and the number of send failures and discarded frames.

NOTE

The UNA and DMP/DMV return line counter information only if a circuit is active on the line.

RSX-11 PSI line counters include such information as bytes and data blocks sent and received, inbound and outbound data errors, and local and remote reply timeouts, buffer errors, and process errors. These counters, together with component characteristics, are useful in monitoring the activity of RSX-11 PSI lines.

For more information on using counters, see Section 2.7. For a complete listing of line counters, see Appendix E.

2.5 Circuits

Circuits are high level communication paths between nodes or DTEs (data terminal equipment) supported by RSX–11 PSI. Circuits are the logical connections between two nodes (or all nodes on an Ethernet cable) that operate over the physical medium of lines.

2.5.1 Circuit Types

DECnet-RSX supports five types of circuits: DDCMP, PCL, Ethernet, DLM, and RSX–11 PSI. Each is briefly described here. Circuit ID formats are outlined later in this chapter.

2.5.1.1 DDCMP Circuits

DDCMP circuits provide the logical connection between two adjacent nodes. They are classified according to the type of line over which they operate:

- A **point-to-point circuit** connects two nodes over a corresponding point-to-point line.
- A **multipoint circuit** operates over a multipoint line, which serves more than two nodes. On a multipoint line, the **multipoint control circuit** is affiliated with the line's control station. In addition, there is one **multipoint tributary circuit** for every tributary on the line.

You can specify multiple circuits from the control (master) end of a control line, with each circuit having a unique physical tributary address. From the tributary (slave) end, you have only one multipoint tributary circuit per line.

2.5.1.2 PCL Circuits

PCL circuits provide point-to-point connections among up to 16 nodes on a line by means of a parallel time division multiplexed bus (TDM). The PCL-11B hardware allocates a fraction of available bus bandwidth to each station.

Each PCL-11B node is configured as a control station. From the perspective of any PCL-11B node on the bus, other PCL-11B nodes appear to be tributaries. However, the concepts of active, dying, and dead tributaries do not apply. Each station is assigned a unique tributary address, which is established when the hardware is installed.

2.5.1.3 Ethernet Circuits

Ethernet circuits provide for multiaccess connection among all of the nodes on the same Ethernet cable. An Ethernet circuit differs from other DECnet circuits in that you connect not to a single node, but to many. Each node on a single Ethernet circuit is considered to be adjacent to every other node on the circuit and equally accessible.

2.5.1.4 DLM Circuits

A data-link-mapping (DLM) circuit is a DECnet circuit that uses RSX-11 PSI lines to connect DECnet nodes. DLM circuits can be one of two types of circuits:

- **Permanent virtual circuit (PVC).** Provides a permanent path between the local DTE and the remote DTE (analogous to a leased line).
- **Switched virtual circuit (SVC).** Provides a temporary path between the local DTE and the remote DTE (analogous to a dial-up line). An SVC is set up using network management commands only when there is data to transmit and is cleared when the line is cleared. The SVC must be designated exclusively for incoming or outgoing calls. Refer to Table 2–6.

2.5.1.5 RSX-11 PSI Circuits

An RSX–11 PSI circuit is a virtual circuit connecting local data terminal equipment (DTE) to a remote DTE. RSX–11 PSI circuits operate differently from DECnet circuits. All RSX–11 PSI circuits pass through the X.25 protocol module (see Section 2.9.1), which multiplexes circuits to lines that it owns. There is no direct relationship between the name of an RSX–11 PSI circuit and an RSX–11 PSI line.

RSX–11 PSI supports the use of both permanent virtual circuits and switched virtual circuits (PVCs and SVCs, respectively). Native RSX–11 PSI SVCs are set up with parameters taken from the X.25 protocol module component when calls are requested on these circuits.

RSX–11 PSI circuits are used in two ways:

- For user application programs; RSX–11 PSI native circuits.
- For the mapping of data link information from the DECnet Routing layer through the X.25 protocol module of the local DECnet/RSX–11 PSI node to the X.25 protocol module of a remote DECnet/RSX–11 PSI node. DLM, which is explained in Section 2.5.5, enables the use of the X.25 protocol for DECnet node-to-node communication.

2.5.2 Circuit Identification

You can use network management commands to modify parameters for an individual circuit or for all known circuits. Use the KNOWN CIRCUITS keyword to specify all circuits identified to the system during network generation including active circuits and circuits in the OFF state. To specify an individual circuit, use the CIRCUIT keyword and a circuit ID. DECnet and RSX–11 PSI circuits have different formats, as described in the following sections.

NOTE

To create a new circuit in the network databases, you must perform another network generation.

2.5.2.1 DECnet Circuit Identification (DDCMP, PCL, Ethernet Circuits)

To specify an individual DECnet circuit, use a circuit ID of the form

dev-c[*-u*][*.t*]

where

dev	is a device name from Table 2–4; for example, DMC, PCL, UNA.
c	is a decimal number, 0 or a positive integer, designating the device's hardware controller.
u	is a decimal unit/circuit number, 0 or a positive integer, included if the device is a multiple line controller.
t	is a decimal number identifying a tributary circuit on a multipoint line.

You can use a wildcard character (*) in place of the controller, unit, or tributary number to specify all known circuits in that category. You can use multiple wildcards in a circuit ID, as long as no numbers follow them. Some examples of DECnet circuit IDs follow:

Circuit ID	Meaning
UNA-1	UNA, controller 1
DZ-0-4.2	DZ, controller 0, unit 4, tributary 2
DZ-0-4.*	DZ, all known tributaries on unit 4 of controller 0
DL-1.3	DL, controller 1, tributary 3
PCL-0.*	PCL, all known tributaries on controller 0

2.5.2.2 DLM Circuit Identification

Circuit IDs for DLM PVCs and SVCs have the form

DLM-*x*.*y*

where

x identifies a group code analogous to the controller number on a device. You can use this to group DLM circuits according to type. For example, all PVCs could have the same group code.

y identifies the circuit number analogous to the logical number associated with tributaries for a device.

2.5.2.3 RSX–11 PSI Circuit Identification

Circuit IDs for RSX–11 PSI PVCs consist of a string of 1 to 6 alphanumeric characters. You cannot identify SVCs for RSX–11 PSI circuits.

2.5.3 Circuit Parameters

The permanent database should contain circuit parameters for all circuits connected to the local node or DTE. You can modify and display circuit parameters by using the following commands which are described in the *DECnet-RSX Guide to Network Management Utilities* :

CFE Commands	NCP/VNP Commands
DEFINE CIRCUIT	SET CIRCUIT
PURGE CIRCUIT	CLEAR CIRCUIT
LIST CIRCUIT	SHOW CIRCUIT

Table 2–6 categorizes all circuit parameters according to circuit type and parameter function.

Table 2–6: Circuit Types and Parameter Functions

Parameters	Circuit Types and Parameter Functions
	All circuits (except RSX–11 PSI)
STATE (CLEARED/OFF/ON/SERVICE)	Sets circuit's operational state
OWNER (DLX/XPT)	Specifies circuit owner
COST[1,2]	Specifies circuit cost for routing
LEVEL 1 COST[1,3,4]	Specifies the level 1 routing cost of the circuit.
LEVEL 2 COST[1,4]	Specifies the level 2 routing cost of the circuit.
HELLO TIMER [1]	Sets frequency of routing on circuit
SERVICE (DISABLED/ENABLED)	Specifies whether down-line loading and loopback testing are enabled
	Multipoint DDCMP circuits
TRIBUTARY	Sets tributary address for circuit
MULTIPOINT ACTIVE	Specifies tributary polling rate

[1]See parameter descriptions in Section 2.2.3.

[2]DECnet–11M/S only.

[3]DECnet–11M–PLUS Level 1 routers only.

[4]DECnet–11M–PLUS area routers only.

Table 2–6 (Cont.): Circuit Types and Parameter Functions

Parameters	Circuit Types and Parameter Functions
	PCL circuits
TRIBUTARY	Sets tributary address for circuit
	Ethernet circuits
MAXIMUM ROUTERS [1] ROUTER PRIORITY [1]	Specifies routing parameters for circuit
	DLM circuits (PVCs or SVCs)
MAXIMUM DATA MAXIMUM WINDOW	Controls data packets for PVCs and SVCs
CHANNEL DTE	Specifies channel and DTE for a PVC
NUMBER	Assigns remote DTE address for an SVC
MAXIMUM RECALLS RECALL TIMER	Controls retransmission when establishing an SVC
USAGE (INCOMING/OUTGOING)	Specifies availability of an SVC for incoming or outgoing calls
	RSX–11 PSI native PVCs
CHANNEL DTE	Specifies channel and DTE for a PVC
COUNTER TIMER	Sets counter timer for event logging
MAXIMUM DATA MAXIMUM WINDOW	Controls data packets for PVCs

[1]See parameter descriptions in Section 2.2.3.

2.5.3.1 Circuit States and Loading

You control the operational state of all circuits at the local node. For DECnet (non-RSX–11 PSI) circuits, setting the circuit state automatically sets the line state as well. This allows you to control circuit traffic and to perform service functions on DDCMP circuits. The state of a circuit can affect the reachability of an adjacent node. Use the STATE parameter to set one or all known circuits to one of the following states:

CLEARED	In the permanent database, this state means that the circuit is not loaded when the network is loaded and is unavailable for use, even though it is defined in the permanent database. In the volatile database or the system image file, this state means that the associated line has not been loaded and is unavailable for use.
OFF	In the permanent database, this state means that the circuit is loaded when the network is loaded, but is not being used. In the volatile database or the system image file, this state means that the circuit is turned off.
ON	In the permanent database, this state means that the circuit is loaded and available for use when the network is loaded. In the volatile database or the system image file, this state means that the circuit is available for use.
SERVICE	This state exists only in the volatile database and the system image file and is valid for DECnet circuits only. It means that the circuit is reserved for service functions such as up-line dumping, down-line system loading, or circuit level loopback testing.

Circuit substates. When you use the NCP SHOW CIRCUIT or SHOW LINE commands explained in Section 3.3.1, the system displays the state to which you have most recently set the circuit(s). In some cases, the system also displays one of the substates listed in Table 2–7.

DECnet software can automatically change the state of a circuit for certain functions. For example, if you initiate a loopback test over a circuit, DECnet network management software automatically changes the state of that circuit to the appropriate internal state or substate.

Several circuit substates have an AUTO- prefix. These substates can occur when you have an adjacent node that is about to be, or is in the process of being, automatically down-line loaded or triggered. For example, if a circuit is in the ON state and the local node senses a request for a down-line load on that circuit, the network management software on the local node automatically sets the circuit to the ON-AUTOSERVICE state.

Table 2–7 lists all circuit states and substates and their meanings. These states and substates also apply to RSX–11 PSI lines. Appendix G lists all RSX–11 PSI circuit, line, and module states. Refer to the *DNA Network Management Functional Specification* for further information about circuit states, substates, and their transitions.

Table 2–7: Circuit/Line States and Substates

State	Substate	Meaning
CLEARED	none	The line has not been loaded.
OFF	none	The circuit/line is not being used.
ON	-RUNNING (not displayed)	The circuit/line is in normal use.
	-STARTING	The circuit is in the (routing) initialization cycle.
	-AUTOSERVICE	The circuit is reserved for automatic service use.
	-LOADING	The circuit is in use for loading.
	-AUTOLOADING	The circuit is in use for automatic loading.
	-DUMPING	The circuit is in use for dumping.
	-AUTODUMPING	The circuit is in use for automatic dumping.
	-TRIGGERING	The circuit is in use for triggering.
	-AUTOTRIGGERING	The circuit is in use for automatic triggering.
	-LOOPING	The circuit/line is in use for active loopback testing.
	-REFLECTING	The circuit/line is in use for passive loopback testing.
	-FAILED	The DLM circuit has unsuccessfully recalled an SVC the specified maximum number of times.
SERVICE	-IDLE (not displayed)	The circuit is reserved for an active service function.
	-LOADING	The circuit is in use for loading.
	-DUMPING	The circuit is in use for dumping.
	-TRIGGERING	The circuit is in use for triggering.
	-LOOPING	The circuit/line is in use for active loopback testing.
	-REFLECTING	The circuit/line is in use for passive loopback testing.

2.5.3.2 Circuit Ownership

On each network node, there are processes that can own the different circuits and control traffic over them. Use the OWNER parameter in the SET CIRCUIT command to set the circuit owner to match the owner for the associated DECnet line. There are two possible owners for DDCMP circuits:

DLX — The Direct Line Access Controller. When DLX owns a circuit, you can use the circuit as an I/O device; you can direct circuit traffic at the Data Link level. The *DECnet-RSX Programmer's Reference Manual* describes the DLX user interface.

XPT — The Routing layer. When XPT owns a circuit, a network user can perform normal logical link operations over it.

If the SERVICE parameter is set to ENABLED, DECnet software can temporarily override XPT as the owner of a DECnet circuit and reserve the circuit for the following service functions:

- Down-line system loading to a remote node
- Up-line dumping from a remote node
- Triggering a remote node
- Loopback testing for a circuit at the physical link level

In these situations, the circuit owner remains XPT, but XPT cannot use the circuit until the service function has finished using the circuit.

Alternatively, DDCMP circuits can be reserved for the service functions listed above by setting the circuit to the SERVICE state in the volatile database. See Section 2.5.3.1.

By default, the owner of DLM circuits is XPT, and the owner of RSX–11 PSI circuits is the NW process. Since Ethernet circuits allow access by multiple users, they do not have exclusive owners. However, the SERVICE parameter can be set to the DISABLED state to disallow any service function requests.

2.5.4 DDCMP Multipoint Circuit Parameters

DECnet-RSX supports DDCMP multipoint circuits for the following devices: DL-11, DLV-11, DMP-11, DMV-11, DU-11, DUP-11, DPV-11, DUV-11, DV-11, DZ-11, KDP-11, and KDZ-11. The following sections describe multipoint operation, including the parameters you can modify to control device operation. Multipoint characteristics are defined at both the line and the circuit level. Applicable parameters are:

For lines:	DEAD TIMER (for DMP and DMV devices only)
	DELAY TIMER (for DMP and DMV devices only)
	MULTIPOINT DEAD
For circuits:	MULTIPOINT ACTIVE
	TRIBUTARY

Circuit parameters apply to a specific tributary. Line parameters apply to all tributaries on the specified line(s).

2.5.4.1 Multipoint Operation

A multipoint circuit has one control station and multiple tributaries. The control station continually polls existing, on-line, tributaries for data awaiting transmission. A tributary is always given the opportunity to respond when it is polled. If the tributary is active, it either transmits data or returns a positive acknowledgment (ACK) if it has no data to transmit. Whenever the control station has data for a tributary, it transmits the data immediately.

An active tributary is one with the circuit turned on. If the control station does not get a response for two consecutive polls, it considers the tributary to be dying and polls less frequently. If the tributary does not then respond to the next six polls, the control station considers it to be dead and polls it even less frequently. If a dead tributary begins to respond, it is reclassified as active and is polled at the original active polling rate.

DECnet software uses polling ratios to control polling rates for active and dead tributaries. The polling ratios have system defaults, but you can use the commands SET/DEFINE CIRCUIT MULTIPOINT ACTIVE and SET/DEFINE LINE MULTIPOINT DEAD, except for DMPs and DMVs, to modify them. For DMPs and DMVs, you must use the DEAD TIMER and DELAY TIMER parameters described later in this section.

The control station polls each active tributary every *n*th time it goes through the polling list (*n* is the active polling ratio set for that tributary). The default ratio is 1, meaning that unspecified active tributaries are polled each time the control station goes through the polling list. There is one dead ratio (default = 8) for all tributaries on a line. The control station polls one dead tributary each *x* times through the polling list (in round robin fashion). The control station polls a dying tributary four times as often as a dead one (or one-fourth the dead polling ratio).

For example, using the defaults, an active tributary is polled once a second, a dying tributary once every two seconds, and one of the dead tributaries every eight seconds. The exact frequency at which a particular tributary is polled depends not only on its active ratio, but also on the states and active ratios of other tributaries on the circuit and the amount of traffic.

To control which tributaries are active, set the related circuit state to ON or OFF. Note that dead tributaries are always polled (according to the dead polling ratio) if they are in the ON state. To improve performance for active tributaries, it is best to turn dead tributaries OFF if they will be dead for a significant period of time.

You must also use the TRIBUTARY parameter to specify an address to be used by the polling mechanism. Correspondingly, you must set the same tributary address for that circuit on the remote node.

2.5.4.2 DMP and DMV Multipoint Controllers

For DMP and DMV controllers, you set the polling rates (in milliseconds) by using the DEAD TIMER and DELAY TIMER parameters in the SET LINE command.

The DEAD TIMER parameter specifies the polling rate for a dead tributary. Start by setting the DEAD TIMER parameter to a value midway between 5000 and 30,000 milliseconds; then adjust the rate to suit your network. If there are no problems with the rate you have set, you may wish to set a higher rate to maximize performance.

The DELAY TIMER, which specifies the maximum time to wait between polls in order to limit the effect of a fast control station on a slow tributary, can have any of the following values:

0	for lines with speeds equal to or greater than 56Kb
50	for multipoint EIA lines with speeds up to 19.2Kb
200	if any DDCMP software implementations are on the line

Always choose the larger value if more than one could apply.

2.5.5 DLM Circuit Parameters

Several circuit parameters that are specific to the operation of DLM circuits are described in the following sections.

2.5.5.1 Remote DTE Addresses

To establish an SVC with a remote DTE, the Routing layer of DECnet software requires the address of the remote DTE. Use the NUMBER parameter in the DEFINE CIRCUIT command to specify the remote DTE address for an outgoing DLM SVC. You should also specify a subaddress as the last digits of the DTE address if your network supports subaddresses. The subaddress that you specify must fall into the subaddress range that is specified for the remote node (see Section 2.1.4.2). If your network does not support subaddresses, you should specify only the DTE address and use the mechanism described in the following paragraphs to determine which incoming calls are intended for DLM circuits.

For outgoing calls, the routing layer uses this address (and the subaddresses required at the remote DTE) to call on the circuit. For example, if the NUMBER parameter was defined using the command

```
CFE> DEFINE CIRCUIT DLM-1 4 NUMBER 3119123456781 RET
```

outgoing calls on circuit DLM-1.4 would call DTE 311912345678 using a subaddress of 1.

For incoming calls, the NUMBER parameter is used to reserve incoming DLM circuits for particular remote DTE addresses. If this parameter is set on an incoming circuit, only calls from the specified remote DTE will be accepted. Do not specify a subaddress on incoming circuits.

So that DLM circuits may be established on networks that do not support subaddressing, incoming calls are treated in two ways:

- If an EXECUTOR SUBADDRESS range is defined, then incoming calls having a subaddress within that range will be treated as DLM calls. All incoming circuits will be searched until one is found that has a NUMBER which matches the calling DTE or until one is found that has no NUMBER

parameter defined. If no such circuit is found, then the call will be cleared. Calls which do not have a subaddress in the EXECUTOR SUBADDRESS range will be handled by the X.25 server module.

- If the EXECUTOR SUBADDRESS range is **not** defined, then all incoming calls will be treated initially as incoming DLM calls. The same search is made of the incoming DLM circuits. However, if a matching circuit is not found, the call is directed to the X.25 server module for comparison against X.25 server destinations. Any calls that have no destination either in DLM circuits or the X.25 server will be cleared.

For example:

```
CFE> DEFINE EXECUTOR SUBADDRESS 1-10 [RET]
CFE> DEFINE CIRCUIT DLM-1.0 NUMBER 311912345600 USAGE INCOMING [RET]
```

causes all incoming calls with a subaddress between 1 and 10 to be treated as DLM calls. Incoming circuit DLM-1.0 will only accept calls from DTE 311912345600.

2.5.5.2 Re-Calls for DLM Circuits

If an attempt to establish a DLM SVC has been unsuccessful, DECnet-RSX attempts to re-call the number. The RECALL TIMER parameter sets the interval that DECnet should wait before attempting a re-call (default: 30 seconds) and the MAXIMUM RECALLS parameter specifies the maximum number of times that DECnet should attempt a re-call (default: 5). You can modify either of these values in a DEFINE CIRCUIT command. For example:

```
CFE> DEFINE CIRCUIT DLM-1.0 RECALL TIMER 20 MAXIMUM RECALLS 10 [RET]
```

If the defined maximum re-call number is exceeded, the circuit is placed in the ON-FAILED state, and you must execute a SET CIRCUIT STATE ON command before another outgoing call can be attempted.

2.5.5.3 DLM Circuit Usage

DLM circuits operate according to the usage you define for them in the permanent database. DLM SVCs may be used either for outgoing or incoming calls. The usage of DLM PVCs is permanent; that is, the circuit is permanently connected to a remote DTE, and does not need to be switched dynamically. The USAGE parameter specifies how DLM circuits are to be used, as in the following example:

```
CFE> DEFINE CIRCUIT DLM-1.4 USAGE OUTGOING [RET]
```

2.5.5.4 Permanent Virtual Circuit Parameters

When PVCs are first specified, the CHANNEL and DTE parameters are mandatory.

The CHANNEL parameter is used to associate a logical channel number with each PVC. This number is allocated to you by the PSDN at subscription time and will be in the range 1 to 4095. Each PVC different from those previously specified for outgoing calls in the SET MODULE X25-PROTOCOL command must have a unique channel number. The command below illustrates the use of this CHANNEL parameter.

```
CFE> DEFINE CIRCUIT DLM-1.6 ... CHANNEL 3 [RET]
```

The DTE parameter is used to associate the local DTE address with each PVC. The command below illustrates the use of the DTE parameter.

```
CFE> DEFINE CIRCUIT DLM-1.6 ... DTE 123789456 [RET]
```

The DTE address is a decimal integer of 1 to 15 digits and must be specified previously in a SET MODULE X25-PROTOCOL command.

The COUNTER TIMER parameter is used to control the frequency with which the circuit counters are logged and zeroed. Circuit counters are described in Section 2.5.7. To stop the logging of circuit counters, use the CLEAR CIRCUIT command.

2.5.5.5 Data Packet Control

Two parameters control the transmission of data packets over the PVC or SVC: MAXIMUM DATA and MAXIMUM WINDOW.

The MAXIMUM DATA parameter specifies the maximum size of the packet for a particular circuit. For example, the following command sets the maximum size of the packet to 128 bytes for the circuit DLM-1.6:

```
CFE> DEFINE CIRCUIT DLM-1.6 ... MAXIMUM DATA 128 [RET]
```

The maximum packet size must always be at least 5 bytes smaller than the maximum size of the frame on a line (see Section 2.4.3.3). Specify a value that is a power of 2 in the range 16 to 4096 bytes.

The MAXIMUM DATA parameter is optional and, by default, takes the network value. See RSX–11 PSI network-specific information for the network value of this parameter.

The MAXIMUM WINDOW parameter specifies the window size for a particular PVC. For example, the command that follows sets the window size to 2 for the circuit DLM-1.6:

```
NCP> SET CIRCUIT DLM-1.6 ... MAXIMUM WINDOW 2 [RET]
```

Specify a value in the range 1 to 127.

The MAXIMUM WINDOW parameter is optional and, by default, takes the network value. See the RSX–11 PSI Network-Specific Information for the network value of this parameter.

2.5.6 RSX–11 PSI Circuit Parameters

If you configured an RSX–11 PSI capability into your system, you can modify certain information that you specified during network generation by using the DEFINE CIRCUIT command with the parameters listed for RSX–11 PSI circuits in Table 2–6. If you use this command to modify the association of a PVC with a local DTE, the same DTE address must be specified in the protocol module. See the CFE DEFINE MODULE X25-PROTOCOL command in the *DECnet-RSX Guide to Network Management Utilities* . Note that both the packet and window size parameters were set to defaults for the PSDN to which the circuit's DTE is connected. Consult the documentation on your specific RSX–11 PSI network for details of these values before you modify them. For more information on window and packet size parameters see Section 2.5.5.5.

RSX–11 PSI channels. All virtual circuits for a DTE are multiplexed over the physical link between the DTE and the network interface (DCE). Each virtual circuit has a logical channel over which data is transmitted at both the DTE and DCE interfaces. Each channel is identified by a unique reference number called a logical channel number (LCN). Each DTE assigns a channel to the virtual circuit and recognizes the virtual circuit only by the LCN that identifies that channel. You use the CHANNELS parameter of the DEFINE MODULE X25-PROTOCOL command to associate the LCN with the DTE.

During network generation, you assigned a channel to each PVC. To modify the logical channel number associated with a circuit in the permanent database, use the CHANNEL parameter in the DEFINE CIRCUIT command. You should not specify an LCN that has already been assigned to an outgoing call in the protocol module. PVC LCNs are assigned by the network vendor when the PVC is set up. More information on the PVC CHANNEL and DTE parameters can be found in Section 2.5.5.4.

2.5.7 Circuit Counters

DECnet-RSX automatically maintains certain statistics, called counters, for circuits in the network. For all circuits, counter information can include the number of messages sent and received over the circuit, timeouts, and the amount of time since the counters were last zeroed. For DDCMP circuits, counters are maintained for timeouts and for data and buffer errors. For both Ethernet and DDCMP circuits, the number of bytes and data blocks sent and received are recorded. For RSX–11 PSI circuits, the counter information includes the time since the counters were zeroed; the number of bytes, data blocks, and resets sent and received; and the number of resets initiated by the network.

For more information on using counters, see Section 2.7. For a complete listing of circuit counters, see Appendix E.

2.6 Event Logging

DECnet-RSX software logs certain network-related events that occur during network operation. The event logger maintains events by logging them at the console, recording events in an event file, or sending them to an event monitor task. Some events that can be recorded are shownin the followng list. A complete list is provided in Appendix D.

- Changes in line, circuit, node, and module states
- Lost events (events that were not logged)
- Service requests (when a line is put in an automatic service state)
- Passive loopback (when the executor is looping back circuit or line level test messages
- Line, circuit, node, and module counter activity

This information can be useful in maintaining and tuning the network because it can be recorded continuously by the event logger. Section 3.4.2 provides an example of a typical event-logging situation with multiple nodes.

As network manager, you are responsible for controlling various aspects of event logging. For details on CFE, NCP, and VNP logging commands, see the following command descriptions in the *DECnet-RSX Guide to Network Management Utilities* :

CFE Commands	NCP/VNP Commands
DEFINE LOGGING	SET LOGGING
PURGE LOGGING	CLEAR LOGGING
LIST LOGGING	SHOW LOGGING

Table 2–8 lists all logging parameters by function and groups them according to operational categories.

Table 2–8: Functions of Logging Parameters

Event Source Parameters	Logging Component Parameters
Event Specification	
EVENTS *event-list*	
KNOWN EVENTS	
Source of Events	
CIRCUIT	
LINE	
MODULE	
X25-PROTOCOL	
X25-SERVER	
X29-SERVER	
NODE	
Logging Component	
	LOGGING CONSOLE
	LOGGING FILE
	LOGGING MONITOR
Logging Component Name	
	NAME

Table 2–8 (Cont.): Functions of Logging Parameters

Event Source Parameters	Logging Component Parameters
Location to Log Events	
	SINK EXECUTOR
	NODE *node-id*
	NODE $HOST
State of Logging Component on the Executor Node	
	STATE OFF
	ON

Following network generation, all events generated by any network component known to the local node are automatically logged at the executor console. You can use the SET LOGGING command to specify that only certain events be logged for certain components: a specified node, line, circuit, or module (X25-PROTOCOL, X25-SERVER, or X29-SERVER).

To remove any or all parameters from the volatile database, use the CLEAR LOGGING command.

2.6.1 Event Specification

Events are defined by class and type. For the most part, events are logged for the various DNA layers and for system-specific resources. You can specify the kinds of events to be logged by using the following event list format with the EVENTS parameter:

class.type[*,type,type,....type*]

where

class identifies the DNA layer or system-specific resource to which the event pertains.

type identifies a particular form of event, unique within an event class. The *type* can be a single number or a range of numbers.

For example, events in the Routing layer are assigned to class 4. The event types for this class range from 0 through 19. Event type 0 indicates aged packet loss, event type 1 indicates unreachable node packet loss, and so forth (see Appendix D for a complete listing of events by class and type). Use the EVENTS parameter to specify a list of events to be logged. To specify logging of all DNA event classes and types that can be generated by a DECnet-RSX node, use the KNOWN EVENTS parameter. When removing logging parameters, use either of the two qualifiers, or specify ALL EVENTS when you want to clear not only all DNA events but also all user-defined events.

When providing an event list for the EVENTS parameter, you can specify only one class, but you can specify multiple event types within a class. You can specify a single event type, a range of types, a combination of these, or a wildcard character (*). The following examples illustrate the format of each.

Event List	Meaning
4.4	Specifies event class 4, type 4.
4.5-7	Specifies event class 4, types 5 through 7.
4.5,7-9,11	Specifies event class 4, types 5, 7 through 9, and 11. Note that types must be specified in ascending order.

2.6.2 Event Sources

Event sources are qualifiers for events. When you specify a source for events, only events generated by that source will be affected. All events have a source type (also known as the entity type) associated with them. There are 6 event entity types:

- Line - the event is generated by a line entity.
- Node - the event is generated by a node entity.
- Area - the event is generated by an area entity.
- Circuit - the event is generated by a circuit entity.
- Module - the event is generated by a module entity.
- None - the event is not associated with any particular network entity.

For example, to monitor network activity over line DPV-0, connected to the local node, you use the following command:

```
NCP> SET LOGGING CONSOLE LINE DPV-0 EVENTS RET
```

Events that pertain to line DPV-0 will now be logged at the console by the event logger. If no source is specified, logging events for all possible sources are affected.

2.6.3 Event Logger Components

CFE, NCP, and VNP commands specify whether events are to be logged to one or all of the following event logger components: a CONSOLE, a FILE, and/or a MONITOR program. A console is any record-oriented device that receives events in ASCII format. A file is a user-specified file that receives events in a DNA machine-readable binary format. The Event File Interpreter program (EVF) which can be used to create reports from this file is described in the *DECnet-RSX Guide to Network Management Utilities* . A monitor program is a user-supplied program that receives and processes those events. Refer to the *DNA Network Management Functional Specification* for a description of the standard DNA event format. Appendix C describes the DECnet-RSX event-logging interface. This information is useful if you want to write a monitor program to receive events.

2.6.4 Logging Component Names

When you initially specify logging components to which event data is to be logged, you can identify them by using the NAME parameter in the SET LOGGING command. Use the CLEAR LOGGING NAME command to clear the name of the logging component and cause events to be sent to the default device for that logging component. The default logging names are:

- For the CONSOLE logging component: CO:
- For the FILE logging component: LB:[1,6]EVENTLOG.SYS
- For the MONITOR logging component: MON...

2.6.5 Logging Sinks

You can use a sink qualifier with any of the logging components for events to be logged at a remote node. Use the SINK parameter to indicate the location of the logging component. The sink can be the executor node, a remote node, or the $HOST node. For example,

```
NCP> SET LOGGING CONSOLE SINK NODE YUKON RET
```

will route all events to the console logging component on the node YUKON. If no sink qualifier is specified, the local node is the default component location. If the sink node is unreachable when the event occurs, the event information is discarded.

2.6.6 Event Logging Component States

You can control the state of each logging component by setting the state of that component to ON or OFF. The OFF state causes all events destined for the logging component to be discarded. The ON state enables logging on the specified logging component. For example,

```
NCP> SET LOGGING CONSOLE STATE OFF RET
```

disables logging to the system operator's terminal on the local node. Note that this does not affect logging to any other logging component whose state may be ON.

2.7 Counters

DECnet-RSX software automatically maintains certain statistics, called counters, for the following network components:

- Circuits
- Lines
- Modules (X.25 protocol and X.25 server)
- Nodes (including the executor)
- System

All counters are listed with a brief description in Appendix E.

Counters are maintained on the node presently designated as the executor and can include such information as the number of data packets sent, received, or lost over a line; system buffer allocation failures; routing packet information; or X.25 protocol message information. Counter statistics are useful alone or when read in conjunction with logging information (see Section 2.6 and Appendix D) to measure and evaluate the performance and throughput of your network configuration.

As system manager, you can display counters and periodically zero them (see Section 2.7) using the NCP SHOW and ZERO commands. In addition, for all circuits and lines, and for RSX–11 PSI modules, you can set a timer whose expiration causes counters for a given component to be logged as an event (see Section 2.7.3). All network management commands for manipulating counters are described in detail in the *DECnet-RSX Guide to Network Management Utilities*

2.7.1 Displaying Counters

You can use the NCP SHOW command. See Section 3.3.1 to display counters for any of the network components listed at the beginning of Section 2.7. The following example shows a sample command and resulting display:

```
NCP> SHOW CIRCUIT DMC-0 COUNTERS RET

Circuit counters as of 23-OCT-86 15:18

Circuit = DMC-0

          >65534  Seconds since last zeroed
            1568  Terminating packets received
            1763  Originating packets sent
         1467600  Transit packets received
         1544950  Transit packets sent
               3  Transit congestion loss
              15  Circuit down
               0  Initialization failure
       102139870  Bytes received
        90378861  Bytes sent
         1606602  Data blocks received
         2023269  Data blocks sent
               2  Data errors inbound, including:
                    NAKs sent, REP response
               0  Data errors outbound
               0  Remote reply timeouts
               0  Local reply timeouts
               0  Local buffer errors
```

Some of these counters can be qualified by information that indicates the condition that contributed to the error. For example, the Data errors inbound counter.

To gain a detailed understanding of counters or the software design and algorithms they represent, consult the appropriate architectural specifications. Refer to Chapter 3 for additional information on the NCP SHOW commands that display network status and counter information.

Several counters in the previous display correspond to network logging events. The events and event descriptions may provide additional information relative to the specific occurrence. Refer to Appendix D for a complete description of events that can be logged. Appendix E maps individual counters to corresponding events where applicable.

2.7.2 Zeroing Counters

When the network is running, you can use the NCP ZERO command to zero any of the network counters. It is wise to zero counters periodically or they will eventually exceed (overflow) their maximum count. Counters that have overflowed display a "greater than" sign (>) in front of their maximum count (see the preceding sample display).

For each component there is a special counter that indicates the number of seconds that have elapsed since the counters for that component were last zeroed. The software increments this counter every second and zeroes it when other counters for the component are zeroed.

2.7.3 Counter Timers and Logging Counters

For lines, circuits, and modules, you can use network management commands to modify the frequency with which counters are logged and then automatically zeroed. To set a timer whose expiration automatically causes counters to be logged on the logging component and then to be zeroed, use the COUNTER TIMER parameter with an NCP/VNP SET CIRCUIT, SET LINE, or SET MODULE command (see the *DECnet-RSX Guide to Network Management Utilities* for command descriptions). For example,

```
NCP> SET LINE KMX-0-0 COUNTER TIMER 600 [RET]
```

causes a line counter logging event to take place every 600 seconds for line KMX-0-0. To clear this parameter, use the following NCP command:

```
NCP> CLEAR LINE KMX-0-0 COUNTER TIMER [RET]
```

This command zeroes the counter timer in the volatile database. You can also set and clear counter timers in the permanent database by using the CFE DEFINE LINE and PURGE LINE commands.

When counters are logged for a particular component as the result of a counter timer expiration, event 0.8 is generated, followed by a listing of the counters.

Example:

```
 Event type 0.8 Automatic counters
Occurred 6-NOV-86 13:17:43 on node 56 (BLTMRE)
Circuit QTPVC
         22 Seconds since last zeroed
          0 Bytes received
          0 Bytes sent
          0 Data block received
          0 Data blocks sent
          0 Locally initiated resets
          0 Remotely initiated resets
          0 Network initiated resets
```

2.8 Processes

DECnet-RSX and RSX-11 PSI processes provide specific functions for the DECnet-RSX/PSI system. For example, the ECL process controls logical link handling, while the PLI process controls X.25 level 3 protocol operations. There are also device driver processes that control communications devices for the system. The RSX operating system is unaware of these processes, except as residents of partitions in system memory.

The following network management commands (described in detail in the *DECnet-RSX Guide to Network Management Utilities* can be used to manipulate processes.

CFE Commands	NCP/VNP Commands
DEFINE PROCESS	SET PROCESS
———	CLEAR PROCESS
LIST PROCESS	SHOW PROCESS

Table 2–9 lists the parameters that can be specified on DEFINE and SET commands.

Table 2–9: Process Parameters

Parameter	DEFINE	SET
ALL		X
LOCATION		X
MAXIMUM CONTROLLERS *count*	X	X
MAXIMUM LINES *number*	X	X
PARTITION *partition-name*		X
STATE	X	

2.8.1 Process Identification

You can use network management commands to modify parameters for an individual process or for all known processes. Use the KNOWN PROCESSES keyword to specify all processes identified to the system during network generation. To specify an individual process, use a process name from Table 2–10. Table 2–10 lists processes alphabetically by type, including device driver processes and software protocol-related processes for DECnet and RSX–11 PSI.

Table 2–10: DECnet-RSX Processes

Process Name	Function
Communications Executive processes	
AUX	Auxiliary process
DLX	Direct Line Access Controller
EVL	Event Logger process
DECnet device drivers	
DHU	DHU-11 driver
DHV	DHV-11 and DHQ-11 driver
DL	DL-11 driver
DLV	DLV-11 driver
DMC	DMC-11/DMR-11 driver
DMP	DMP-11 driver
DMV	DMV-11 driver
DPV	DPV-11 driver
DU	DU-11 driver
DUP	DUP-11 driver
DUV	DUV-11 driver
DV	DV-11 driver
DZ	DZ-11 driver
DZV	DZV-11 driver
KDP (KMC/DUP)	KDP-11 driver
KDZ (KMC/DZ)	KDZ-11 driver
PCL	PCL driver
UNA	DEUNA/DELUA driver
QNA	DEQNA/DELQA driver

(continued on next page)

Table 2–10 (Cont.): DECnet-RSX Processes

Process Name	Function
DECnet processes	
DCP	Implements DDCMP; included for all systems with devices other than DMC-11, DMP-11, DMR-11, DMV-11, PCL-11B, DEUNA, DEQNA, DELUA, and DELQA
EPM	Ethernet Protocol Manager; included for all systems with Ethernet devices
ECL	End Communications layer process
XPT	Routing layer process
NCT	Network Command Terminal process
RTH	Remote Terminal Host process
LAT	LAT device driver process
RSX–11 PSI device drivers	
KMX	KMX-11 driver
KMY	KMY-11 driver
KMV	KMV-11 driver
SDP	DUP-11 driver
SDV	DPV-11 driver
RSX–11 PSI processes	
DLM	Data-link-mapping process
LAB	X.25 level 2 LAPB protocol process
NW	RSX–11 PSI user interface process
PLI	X.25 level 3 (packet level) process

2.8.2 Process States and Loading

During network generation, the required processes are set up in the permanent database and are automatically loaded when you execute an NCP SET SYSTEM command. Processes for lines and circuits can be automatically loaded when you issue commands in one of the following formats:

```
CFE> DEFINE LINE line-id STATE ON

NCP> SET LINE line-id ALL
```

A process remains in memory until you clear the system or the process.

To load a process that has not been automatically loaded into the volatile database, use the NCP SET PROCESS ALL command. This loads the identified process with the default information specified in the permanent database for each process parameter.

The LOCATION parameter allows you to specify the memory location into which to load a process when it is manually loaded. This enables you to manage memory allocation for the system. The TOPDOWN option causes the software to be loaded from the top of the partition down. The FIRSTFIT option causes the first available space in the partition to be used.

The PARTITION parameter allows you to specify an alternative partition other than the default partition.

2.8.3 Maximum Controllers and Lines

During network generation, you specify the maximum number of lines and controllers that each process can control. If necessary, you can use the MAXIMUM CONTROLLERS and MAXIMUM LINES parameters to modify this information. These parameters determine the amount of additional space to be reserved in the process partition that is used to allocate process tables. If more lines are loaded than specified by the MAXIMUM LINES parameter, the additional line tables are allocated from the system pool.

2.9 RSX–11 PSI Modules

There are four RSX–11 PSI modules that can be manipulated by DECnet-RSX network management commands:

- **X.25 protocol module.** Identifies local DTE addresses and group names
- **X.25 server module and X.29 server module.** Identifies destinations
- **X.25 access module.** Replaces remote DTE addresses with logical destination names.

Each module, its function, and its parameters are described in the following sections. The following network management commands (described in detail in the *DECnet-RSX Guide to Network Management Utilities*) can be used to control the four RSX–11 PSI modules.

CFE Commands	NCP/VNP Commands
DEFINE MODULE	SET MODULE
PURGE MODULE	CLEAR MODULE
LIST MODULE	SHOW MODULE

2.9.1 X.25 Protocol Module

The X.25 protocol module implements the X.25 level 3 protocol that controls the transmission of data packets. In particular, this module structures control and user data into packets, sequences these packets for transmission, and establishes, maintains, and clears RSX–11 PSI virtual circuits. This module also associates local DTE addresses and possibly group names with this controlling information.

The X.25 protocol parameters consist of three types: local DTE-related parameters, group-related parameters, and protocol-related control parameters. Table 2–11 lists all protocol module parameters by type that you can modify, and indicates the network management utilities that operate on each parameter.

Table 2–11: Protocol Module Parameters

Parameter	CFE	NCP
Local DTE-related parameters		
DTE *dte-address*	X	
CHANNELS *list*	X	
COUNTER TIMER *seconds*	X	X
LINE *line-id*	X	
MAXIMUM CIRCUITS *count*	X	
STATE	X	X
Group-related parameters		
GROUP *group-name*	X	X
DTE *dte-address*	X	X
NUMBER *group-number*	X	X
TYPE BILATERAL	X	X
Protocol-related parameters		
MAXIMUM DATA *byte-count*	X	
DEFAULT DATA *byte-count*	X	
MAXIMUM WINDOW *count*	X	
DEFAULT WINDOW *block-count*	X	
CALL TIMER *seconds*	X	
CLEAR TIMER *seconds*	X	
MAXIMUM CLEARS *count*	X	
RESET TIMER *seconds*	X	
MAXIMUM RESETS *count*	X	
RESTART TIMER *seconds*	X	
MAXIMUM RESTARTS *count*	X	

The following sections describe these parameters and illustrate the use of network management commands for modifying X.25 protocol module parameters.

2.9.1.1 Local DTE-Related Parameters

Every DTE in the network is identified by one or more local DTE addresses. In order to modify parameters for local DTEs, you need to identify the DTE address in the network management command. You can modify parameters for a specific DTE or for all known DTEs. Known DTEs include all local DTEs identified to the system during network generation. Use the KNOWN DTES qualifier to identify all known DTEs for which parameters are to be modified. To identify an individual DTE, use the DTE qualifier with an integer address consisting of 1 to 15 digits. For example;

```
CFE> DEFINE MODULE X25-PROTOCOL DTE 123456789 RET
```

Local DTE-related parameters include the following:

- The local DTE address
- A set of logical channel numbers for the local DTE
- A timer for automatically logging protocol module counters on a per DTE basis
- The line associated with the local DTE
- The maximum number of circuits that the local DTE may use
- The operational state of the local DTE

You can modify all of these parameters except the local DTE's state and counter timer only in the permanent database.

2.9.1.2 Logical Channels

Use the CHANNEL parameter to identify the set of logical channels associated with the DTEs that are to be used for outgoing calls. Outgoing calls are all calls that originate from your DTE. Specify one or more logical channel numbers (LCNs) as a list. Separate multiple LCNs with hyphens to indicate ranges, and commas to indicate single numbers. For example,

```
CFE> DEFINE MODULE X25-PROTOCOL DTE 123456789 CHANNELS 20-10,8,3 RET
```

indicates that 20 is the first LCN in a range to 10, then 8, and finally 3.

The CHANNELS parameter is mandatory when you specify a DTE for the first time. The values you should supply are those supplied by the network authorities at subscription time. Channel values should be specified in descending order.

You can modify this parameter only in the permanent database.

2.9.1.3 DTE Counters

Use the COUNTER TIMER parameter to alter the frequency with which the DTE counters are logged and zeroed. To stop DTE counters from being logged use the CLEAR MODULE command.

2.9.1.4 Local DTE Line

Use the LINE parameter to identify the line associated with the DTE. Each local DTE must be associated with a unique X.25 physical line. Specify a *line-id* in the form *dev-c-u* (for more information on line identification, see Section 2.4.2). For example,

```
CFE> DEFINE MODULE X25-PROTOCOL DTE 123456789 LINE KMX-1-0 RET
```

This example specifies that line KMX-1-0 will be used for all outgoing calls for DTE 123456789. If you change the line associated with the DTE, you may need to change certain line parameters (Section 2.4.3).

The LINE parameter is mandatory when you are specifying a DTE for the first time.

You can modify this parameter only in the permanent database.

2.9.1.5 Maximum Circuits

Use the MAXIMUM CIRCUITS parameter to specify the maximum number of circuits that can be in use at any one time for the DTE. This count includes circuits for both incoming and outgoing calls. For example,

```
CFE> DEFINE MODULE X25-PROTOCOL DTE 123456789 MAXIMUM CIRCUITS 30 RET
```

specifies that DTE 123456789 can handle at most 30 simultaneous circuits.

The MAXIMUM CIRCUITS parameter is optional and, by default, is set to 255.

You can modify this parameter only in the permanent database.

2.9.1.6 Local DTE States

Use the STATE parameter to specify the operational state of the DTE. There are three states associated with a local DTE:

OFF — In the permanent database, this state means the DTE is not available for use when the system is loaded. In the running database or system image file, this state means that the DTE cannot be used. All existing virtual circuits are cleared.

ON — In the permanent database, this state means the DTE is available for normal use once the system is loaded. In the running database or system image file, this state means that the DTE is available for normal use.

SHUT — This state exists only in the running database and system image file. It means no new virtual circuits may be made with the DTE, but existing virtual circuits are not terminated. When all circuits terminate, the DTE goes to the OFF state. All existing SVCs are cleared immediately. PVCs are not cleared immediately; however, a no-com indication is issued to each PVC to warn that the DTE's state has changed.

The ON and SHUT states have substates; see Appendix G for RSX-11 PSI module states, substates, and transitions. The STATE parameter is optional; if it is not specified the default state is ON.

2.9.1.7 Group-Related Parameters

At subscription time, you may have chosen an optional user group facility to restrict DTE communication. Groups can be bilateral closed user groups (BCUGS) or closed user groups (CUGS). In order to modify parameters for a group, you must identify the group name in the network management command. You can modify parameters for an individual group or for all known groups. Known groups include all groups identified to the system during network generation. Use the KNOWN GROUPS qualifier to identify all known groups for which parameters are to be modified. To identify an individual group, use the GROUP qualifier with a group name consisting of one to six alphanumeric characters. For example,

```
CFE> DEFINE MODULE X25-PROTOCOL GROUP CUGBOS [RET]
```

Group-related parameters include the following:

- The name of the group associated with a local DTE
- The group number and type

All group-related parameters can be modified both in the permanent and running databases.

2.9.1.8 Group DTE Identification

Every group name must be associated with a local DTE address. Use the DTE parameter to specify a DTE address for the group name. For example,

```
CFE> SET MODULE X25-PROTOCOL GROUP CUGBOS DTE 123456789 RET
```

associates DTE 123456789 with group CUGBOS.

The DTE parameter is mandatory when you specify a group for the first time.

2.9.1.9 Group Number

Use the NUMBER parameter to modify the group number. Group numbers are allocated at subscription time. They are specified as one to two digits for CUGS, and one to four digits for BCUGS.

The NUMBER parameter is mandatory when you specify a group for the first time.

2.9.1.10 Group Type

Use the TYPE parameter only to indicate whether the group is a bilateral closed user group (BCUG). BILATERAL is the only keyword for the TYPE parameter. If your group is not bilateral, do not use this parameter.

2.9.1.11 Protocol-Related Parameters

During network generation, you specified a number of protocol-related parameters that affect the operation of the X.25 protocol module. Protocol-related control parameters include the following:

- Data packet control parameters including the default and maximum packet size and window size
- Call request packet control
- Clear request packet control parameters including a clear timer and the maximum number of clears
- Reset control parameters including a reset timer and the maximum number of resets

- Restart control parameters including a restart timer and the maximum number of restarts

You can modify these parameters only in the permanent database.

Note that both the packet and window size parameters were set to defaults for the PSDN to which the DTE is connected. Consult the documentation on your specific RSX-11 PSI network for details of these values before you modify them.

2.9.1.12 Packet Size

Use the MAXIMUM DATA parameter to specify the maximum size of a packet for all SVCs. Specify a size at least five bytes smaller than the maximum size of a frame on a line (see Section 2.4.3).

Use the DEFAULT DATA parameter to specify the default size of a packet for all SVCs. You should have agreed upon a default packet size with the PSDN vendor. Specify a size no greater than the maximum block size.

For example,

```
CFE> DEFINE MODULE X25-PROTOCOL - RET
CFE> MAXIMUM DATA 128 DEFAULT DATA 64 RET
```

specifies a default packet size of 64 and a maximum packet size of 128 for all SVCs. The command is shown here in continuation format.

2.9.1.13 Window Size

Use the MAXIMUM WINDOW parameter to specify the maximum size of a window for all SVCs. The maximum window size defines how many data packets you may send before you have to wait for an acknowledgment.

Use the DEFAULT WINDOW parameter to specify the default size of a window for all SVCs. You should have agreed upon a default window size with the PSDN vendor. Specify a size no greater than the maximum window size. For example;

```
CFE> DEFINE MODULE X25-PROTOCOL - RET
CFE> MAXIMUM WINDOW 3 DEFAULT WINDOW 2 RET
```

specifies a default window of 2 and a maximum window of 3 for all SVCs. The command is shown here in continuation format.

2.9.1.14 Call Request Timer

Use the CALL TIMER parameter to modify the way call requests are controlled. The call timer starts when a request to set up a virtual circuit is transmitted. If no response is received within the time specified, the request is cleared. You can specify the time as follows. For example,

```
CFE> DEFINE MODULE X25-PROTOCOL CALL TIMER 10 [RET]
```

clears the request to set up a virtual circuit if no response has been received within 10 seconds.

2.9.1.15 Clear Request Timer and Maximum Clears

Use the CLEAR TIMER and MAXIMUM CLEARS parameters to modify the way clear requests are controlled. The CLEAR TIMER parameter specifies how long the software waits for an acknowledgment of a request to clear a virtual circuit before retransmitting the request. The MAXIMUM CLEARS parameter specifies how many times the request can be retransmitted. For example,

```
CFE> DEFINE MODULE X25-PROTOCOL CLEAR TIMER 10 MAXIMUM CLEARS 8 [RET]
```

means if a request to clear a virtual circuit is not acknowledged within 10 seconds, the request is retransmitted. This operation may be repeated up to eight times.

2.9.1.16 Reset Timer and Maximum Resets

Use the RESET TIMER and MAXIMUM RESETS parameters to modify the way resets are controlled. The RESET TIMER parameter specifies how long the software waits for an acknowledgment of a reset before transmitting another reset. The MAXIMUM RESETS parameter specifies how many resets can be transmitted. For example,

```
CFE> DEFINE MODULE X25-PROTOCOL RESET TIMER 10 MAXIMUM RESETS 10 [RET]
```

means if a reset is not acknowledged within 10 seconds another reset is transmitted. This operation may be repeated up to 10 times. If no acknowledgment is received, a clear request is transmitted.

2.9.1.17 Restart Timer and Maximum Restarts

Use the RESTART TIMER and MAXIMUM RESTARTS parameters to modify the way restarts are controlled. The RESTART TIMER parameter specifies how long the software waits for an acknowledgment of a restart before transmitting another restart. The MAXIMUM RESTARTS parameter specifies how many restarts can be transmitted. For example:

```
CFE> DEFINE MODULE X25-PROTOCOL RESTART TIMER 20 - RET
CFE> MAXIMUM RESTARTS 6 RET
```

means if a restart is not acknowledged within 20 seconds another restart is transmitted. This operation may be repeated up to six times. If no acknowledgment is received, a clear request is transmitted.

2.9.2 X.25 Server and X.29 Server Modules

The way the software handles incoming calls is determined by parameters you define for X.25 server and X.29 server module destinations. The RSX–11 PSI software uses X.25 server module parameters to handle incoming calls for the RSX–11 PSI user program interface. It uses X.29 server module parameters to handle incoming calls for the RSX–11 PSI X.29 terminal interface. Section 2.9.2.1 describes the process of incoming call handling for nodes with an RSX–11 PSI capability.

There are two types of server module parameters: server-related parameters and destination-related parameters. Server-related parameters affect the operation of the server software, while destination-related parameters define information for handling incoming calls.

Table 2–12 lists, by type, all server module parameters that you can modify and indicates the network management utilities that operate on each parameter.

Table 2–12: Server Module Parameters

Parameter	CFE	NCP
Destination-related parameters		
CALL MASK *hex-value*	X	X
CALL VALUE *hex-value*	X	X
GROUP *group-name*	X	X
NUMBER *dte-address*	X	X
SUBADDRESSES *range*	X	X
OBJECT *object-id*	X	X
PRIORITY *priority*	X	X
Server-related parameters		
COUNTER TIMER *seconds*	X	X
MAXIMUM CIRCUITS *count*	X	
STATE		X *

* You cannot specify STATE for the X.29 server module

The following sections describe parameters noting differences where they arise. These sections also illustrate the use of network management commands for modifying server module parameters.

NOTE

You cannot modify parameters for existing destinations. You can only create new destinations. Existing destinations may be cleared then reset with new parameters. Also, if a destination name is specified, the object also must be specified.

2.9.2.1 Incoming Call Handling

Whenever a remote DTE attempts to communicate with your local DTE (that is, attempts to set up a virtual circuit), both the remote DTE and the local DTE provide information that identifies the destination of the call. Remote DTE information passed with the call may include the remote DTE address, a local DTE subaddress, possibly a closed user group (CUG) or bilateral closed user group (BCUG) name, and possibly a value in the user data field. The RSX–11 PSI software at the local DTE uses this information along with destination information defined in the running database for the server module to determine how to handle the incoming call. This section describes the process of incoming call handling as it relates to network management and parameters that you specify in the network databases. The *RSX–11 PSI User's Guide* describes incoming call handling as it relates to programming RSX–11 PSI network tasks.

When an incoming call is detected, the network software constructs a network information block (NIB) using the remote DTE address, the local DTE subaddress, and, if specified, the CUG or BCUG name and user data. The RSX–11 PSI software then attempts to match the information in these fields with the information specified for destinations in the running database.

If only one match is found, the software associates the incoming call with the object specified for this destination. If more than one match is found, the software chooses the destination with the highest priority. Then, it associates the incoming call with the object specified for this destination.

The object names the task which is to run when the incoming call activates it. If the task is multicopy, a new instance of the task is created for each incoming call. Section 2.9.2.7 discusses objects and object parameters.

The software rejects the incoming call if no match is found and a last chance handler has not been specified. A last chance handler is a destination with an associated object that handles all incoming calls for which a match cannot be found. The simplest last chance handler is a destination that specifies the complete range of local DTE subaddresses. The last chance handler must have the lowest priority of all the destinations.

2.9.2.2 Destination-Related Parameters

In order to specify parameters for a destination, you must identify the destination in the network management command. You can modify parameters for an individual destination or for all known destinations. Known destinations include all destinations identified to the system during network generation. Use the KNOWN DESTINATIONS qualifier to identify all known destinations for which parameters are to be modified. To identify an individual destination, use the DESTINATION qualifier with a destination name consisting of 1 to 6 alphanumeric characters. For example,

```
CFE> DEFINE MODULE X25-SERVER DESTINATION AKRON [RET]
```

There are five destination parameters that you can modify to determine whether a destination can accept an incoming call. In addition, there are two remaining parameters you can use to specify the priority of the destination and the task to which the destination passes the call for processing.

Destination-related parameters include the following:

- The name of the destination
- A call mask for testing incoming calls
- A call value for testing incoming calls
- The name of the group associated with the destination
- A range of subaddresses for the destination
- The identification of the task to be activated
- The priority of the destination

2.9.2.3 Call Value and Call Mask (User Data Field)

Use the CALL VALUE and CALL MASK parameters to modify information in the user data field of a destination. The destination, by first applying the call mask to the user data field of the incoming call, and then comparing user data with the call value, decides if it can accept the call. The call value and the call mask must be the same length. The Comite Consultatif International Telephonique et Telegraphique (CCITT) recommends that you use a call data field value of 01 for incoming X.29 calls. For example,

```
NCP> SET MODULE X29-SERVER DESTINATION CHI CALL MASK FF -[RET]
NCP> CALL DATA 01 [RET]
```

causes the mask of FF to be logically ANDed with the user call data and then a comparison between the user call data and 01 to occur. The call will be accepted only if the user data matches the value 01.

The CALL MASK and CALL VALUE parameters are optional and, by default, no mask or value is used to determine if the destination can handle an incoming call.

2.9.2.4 Group Identification

Use the GROUP parameter to modify the name of the group whose calls are handled by a destination. You should ensure that this name is the same as the group name specified in the protocol module (Section 2.9.1.7). An example of a command using the GROUP parameter is as follows:

```
NCP> SET MODULE X29-SERVER DESTINATION CHI GROUP BCUG [RET]
```

This example causes the destination CHI to accept only calls with a group name of BCUG.

The GROUP parameter is optional and, by default, no group name is used to determine if the destination can accept an incoming call.

2.9.2.5 Remote DTE Identification

Use the NUMBER parameter to modify the address of the remote DTE that can send calls to a destination. For example,

```
NCP> SET MODULE X29-SERVER DESTINATION CHI NUMBER 123456 [RET]
```

specifies that destination CHI will only accept calls from the remote DTE with an address of 123456.

The NUMBER parameter is optional and, by default, no remote DTE address is used to determine if the destination can accept an incoming call.

2.9.2.6 Subaddresses

Use the SUBADDRESSES parameter to modify the range of subaddresses for a destination. Only those incoming calls that specify subaddresses in the range you specify are accepted by the destination. For example,

```
NCP> SET MODULE X26-SERVER DESTINATION CHI SUBADDRESS 12-24,35 [RET]
```

specifies that all incoming calls with a subaddress value of 12 to 24 or 35 will be accepted by destination CHI.

The SUBADDRESSES parameter is optional and, by default, no subaddress range is used to determine if the destination can accept an incoming call. Subaddresses should be entered in ascending order.

2.9.2.7 Object Identification

Use the OBJECT parameter to modify the name or number of the task that is activated when the destination accepts an incoming call. Specify a task name, or, for X.25 multicopy tasks, specify an object number. For X.29 destinations, you must use a task name. For example,

```
NCP> SET MODULE X29-SERVER DESTINATION CHI OBJECT XTR [RET]
```

causes all calls for destination CHI to be routed to the XTR task.

An object parameter must be specified when setting up a destination.

2.9.2.8 Priority

Use the PRIORITY parameter to modify the priority of a destination. If an incoming call can be accepted by more than one destination, the destination with the highest priority that is free to handle the call is used. For example,

```
NCP> SET MODULE X29-SERVER DESTINATION CHI PRIORITY 10 [RET]
```

causes destination CHI to have a priority of ten.

The PRIORITY parameter is mandatory if you specify more than one destination to handle the same incoming call. Otherwise, the parameter defaults to zero.

2.9.2.9 Server-Related Parameters

Server-related parameters include the following:

- The maximum number of circuits that the module may use (that is, all destinations plus all tasks making outgoing calls)
- A timer for automatically logging server module counters
- The operational state of the server module (X.25-only)

2.9.2.10 Maximum Circuits

During network generation, you specified the maximum number of circuits that each server module could handle simultaneously. This count includes circuits for both incoming and outgoing calls. You can modify this information in the permanent database.

Use the MAXIMUM CIRCUITS parameter to modify this count for each server module. For example,

```
CFE> DEFINE MODULE X25-SERVER MAXIMUM CIRCUITS 30 [RET]
```

defines an X.25 server module that can handle 30 circuits simultaneously.

The MAXIMUM CIRCUITS parameter is optional and, by default, the maximum is set to 255.

2.9.2.11 X.25 Server Module Counters

Use the COUNTER TIMER parameter to specify the frequency with which the server module counters are logged and zeroed. More information on module counters can be found in Section 2.9.4. To stop the logging of counters, use the CLEAR MODULE command.

2.9.2.12 X.25 Server Module States and Loading

The X.25 server module has three states in the running database:

OFF	allows no virtual circuit activity, terminates existing circuits, and forces the release of all mailboxes immediately.
ON	allows normal virtual circuit activity.
SHUT	allows no new virtual circuits, does not destroy existing virtual circuits, and goes to the OFF state when all virtual circuits terminate and all mailboxes are released.

Use the STATE parameter of the SET MODULE X25-SERVER command to change the state of the module. The states and state transitions for RSX–11 PSI modules are given in Appendix G.

The STATE parameter is optional and, by default, is set to ON. You cannot specify the STATE parameter for the X.29 server module.

2.9.3 X.25 Access Module

You can use the X.25 access module to create RSX–11 PSI logical destination names to use as you would alias node names (see Section 2.1.1.2) in place of a remote DTE address whenever you identify a remote DTE in a PSDN. RSX–11 PSI software handles the translation from logical destination name to remote DTE address. You can create or modify logical destination names in any of the three databases. For example, to create a logical destination name in the volatile database, use the NCP SET MODULE X25-ACCESS command (shown here in continuation format):

```
NCP> SET MODULE X25-ACCESS DESTINATION STEVE NUMBER 12356 - RET
NCP> TERMINAL TT11: RET
```

This command specifies that logical destination name STEVE can be used in place of remote DTE number 12356 for network user programs initiated from terminal TT11:. You can specify whether the logical destination name applies to a specific terminal (TERMINAL *term-id*), as in this example, or to all terminals (GLOBAL). The default scope is your terminal.

2.9.4 RSX-11 PSI Module Counters

RSX-11 PSI automatically maintains certain statistics, called counters, for the X.25 protocol, X.25 server, and X.29 server modules. For the X.25 protocol module, these counters include the number of bytes, data blocks, calls, and fast selects sent and received; the number of active channels; the number of resets sent, received, or initiated by the network; and the number of restarts. These statistics are useful in monitoring the activity of the component.

X.25/X.29 server module counters include the maximum number of active circuits, the number of incoming calls rejected, the time since the counter was last zeroed, and a counter timer value. For more information on using counters, see Section 2.7. For a complete listing of module counters, see Appendix E.

2.10 System Component

The permanent database contains information about the system component that controls the way the Communications Executive is set up and loaded.

2.10.1 System Parameters

You can modify and display system component parameters by using the following commands (which are described in the *DECnet-RSX Guide to Network Management Utilities*):

CFE Commands	NCP/VNP Commands
DEFINE SYSTEM	SET SYSTEM
———	CLEAR SYSTEM
LIST SYSTEM	SHOW SYSTEM

Note that you cannot remove system parameters from the permanent database. The system parameters you can specify are summarized in Table 2–13.

Table 2–13: System Parameters and Their Functions

Parameters	Parameter Functions
LARGE BUFFER SIZE	Specify buffer size
MAXIMUM CONTROL BUFFERS	Specify number of buffers
MAXIMUM LARGE BUFFERS	
MAXIMUM SMALL BUFFERS	
MINIMUM RECEIVE BUFFERS	
LOCATION	Specify network software location
POOL BYTE-AREA	
POOL PARTITION	

2.10.1.1 System Buffers

Five parameters control the number and size of system buffers. See Chapter 6 for a full description of system buffers and DECnet-RSX memory usage.

Large Buffers. DECnet-RSX uses large buffers for intermediate storage of all user data being sent (transmit buffers) and all incoming messages (receive buffers). Use the LARGE BUFFER SIZE parameter to change the size of the large buffers, and the MAXIMUM LARGE BUFFERS parameter to specify the maximum number of large buffers available for system use. For example:

```
CFE> DEFINE SYSTEM LARGE BUFFER SIZE 256 MAXIMUM LARGE BUFFERS 40 RET
```

Control Buffers. DECnet-RSX uses control buffers to pass parameters between processes. Use the MAXIMUM CONTROL BUFFERS parameter to specify the maximum number of control buffers available for system use.

Small Buffers. DECnet-RSX uses small buffers to store shorter messages where large data buffers would waste memory space. Usually these messages are internal protocol control messages. Use the MAXIMUM SMALL BUFFERS parameter to specify the maximum number of small buffers available for system use.

Receive Buffers. DECnet-RSX reserves a number of the large buffers for use as receive buffers. This enables you to give receive traffic priority over transmit traffic, since the system will service requests for receive buffers first, up to the minimum specified. Use the MINIMUM RECEIVE BUFFERS parameter to specify the minimum number of receive buffers available for system use.

2.10.1.2 System Pool

Two parameters control the size and location of the system network pool.

Pool Area. Use the POOL BYTE-AREA parameter to specify the number of bytes required for the network pool byte area. For example:

```
CFE> DEFINE SYSTEM POOL BYTE-AREA 4096 [RET]
```

Pool Partition. Use the POOL PARTITION parameter to specify the name of the partition into which the network pool will be loaded. For example:

```
CFE> DEFINE SYSTEM POOL PARTITION PSIPL [RET]
```

2.10.1.3 System Location

Use the LOCATION parameter to specify where the system process will be loaded in memory. Two options are available:

FIRSTFIT	loads the processes at the first available space that is large enough for the complete system.
TOPDOWN	loads the processes at the top of the partition.

3
Operating DECnet-RSX/PSI Nodes

This chapter provides the basic information you need to perform the following functions:

- Start up and shut down a local DECnet-RSX node: Section 3.1
- Start up and shut down a local DECnet-RSX/PSI node: Section 3.2
- Monitor local and remote nodes within a DECnet-RSX/PSI network: Section 3.3

Specific CFE, NCP, and VNP commands are defined throughout this chapter. See the *DECnet-RSX Guide to Network Management Utilities* for more information on using these commands.

3.1 Controlling a Local DECnet-RSX Node

The following two sections contain the procedures for starting up and shutting down a DECnet-RSX node.

3.1.1 Starting Up a Local DECnet-RSX Node

The start-up procedure for a local node loads the Communications Executive (CEX) into memory and activates the DECnet-RSX software. If your DECnet system includes RSX-11 PSI, RSX-11 PSI is automatically loaded, but you must still start it up. See Section 3.2.1.

To load the DECnet software into memory, use one of the following methods:

- The NETINS.CMD file for newly generated systems
- NCP commands for previously installed software
- VNP commands to reboot from the system image file

3.1.1.1 Using the NETINS.CMD File

The NETINS.CMD file is a product of NETGEN and is described in detail in the *DECnet-RSX Network Generation and Installation Guide*. After running NETGEN, you must use NETINS.CMD to start up a newly generated DECnet-RSX system for the first time. To load a DECnet-RSX system using this file, enter

```
> @[grp,1]NETINS
```

where *grp* is the network UIC group code selected at NETGEN.

3.1.1.2 Using NCP Commands

Use the following NCP commands to load and start up a DECnet-RSX system when the network software has been installed (by a previous run of NETINS.CMD, for example). Any lines and circuits that are in the ON state in the permanent database will be loaded and turned on as a result of executing these commands.

```
NCP> SET SYSTEM [RET]
NCP> SET EXECUTOR STATE ON [RET]
```

3.1.1.3 Using VNP Commands

Use the following VNP commands to incorporate the DECnet-RSX software components into the RSX-11 system image file. You must perform an RSX SAV on the system image file before you run VNP. Once you have run VNP, DECnet-RSX will start up automatically the next time you boot the RSX operating system.

```
> VNP [RET]
Enter filename: RSX11S.SYS [RET]
VNP> SET SYSTEM [RET]
VNP> SET EXECUTOR STATE ON [RET]
VNP> SET MODULE X25-SERVER [RET]
```

NOTES

1. On RSX-11M/M-PLUS systems, you must have installed NTINIT using VMR before you can execute a VNP SET SYSTEM command.
2. On RSX-11M/M-PLUS systems, you cannot use the VNP SET EXECUTOR STATE ON command; this function must be performed at start-up time using the NCP SET EXECUTOR STATE ON command.

3.1.2 Loading and Turning On a Line in a Running System

After you have started up the system, you can still load a line, or all lines known to the system, and turn on the associated circuit(s) by issuing NCP commands using one of the following pairs of formats:

SET LINE *line-id* ALL
SET CIRCUIT *circuit-id* STATE ON

SET KNOWN LINES ALL
SET KNOWN CIRCUITS STATE ON

All lines to be loaded must be in the CLEARED state. The ALL parameter specifies that the line(s) are to be loaded with the default parameter values specified in the permanent database. If you wish to change the defaults, you must specify the appropriate SET LINE command parameters. See Section 2.4.3 for more information.

To cause a new line or all known lines to be loaded and turned on the next time the system is started up, use the CFE DEFINE versions of the preceding commands to set up the line(s) and circuit(s) in the permanent database. The line(s) will be automatically loaded and their circuit(s) turned on the next time you issue the SET SYSTEM and SET EXECUTOR STATE ON commands.

3.1.3 Shutting Down DECnet-RSX

You can shut down selected lines or the entire node, as described in the following sections.

3.1.3.1 Shutting Down DECnet-RSX Circuits and Lines

You shut down a DECnet circuit by setting the state of the circuit to OFF. You can shut down one or all DECnet circuits by issuing NCP commands using one of the following formats:

SET CIRCUIT *circuit-id* STATE OFF

SET KNOWN CIRCUITS STATE OFF

After shutting down all the circuits on a given line, you can cause the associated line to be unloaded from the system by issuing NCP commands using one of the following formats:

CLEAR LINE *line-id* ALL

CLEAR KNOWN LINES ALL

Before using the line again you must reload the line using the SET LINE command.

3.1.3.2 Shutting Down a DECnet-RSX Node

Use the NCP SET EXECUTOR STATE command to shut down node activity immediately or in an orderly fashion, depending upon the state you specify.

State	Result
SHUT	**Orderly shutdown.** Prevents any new logical links to or from the node, but does not cut off currently active links. After all links have disconnected, the system returns event message 2.0, stating that the executor state is OFF.
OFF	**Immediate shutdown.** Immediately terminates all network activity without allowing active links to disconnect in an orderly manner. All active links are aborted and active tasks are notified of network shutdown.

After shutting down the node, you can unload any remaining network software using the NCP CLEAR SYSTEM command.

You can also initiate the orderly shutdown of a DECnet-RSX node, and the unloading of any remaining network software, by executing the NETREM.CMD command file:

```
> @[grp,1]NETREM
```

where *grp* is the network UIC group code selected at NETGEN. For more information, see the *DECnet-RSX Network Generation and Installation Guide*.

NOTE

The SHUTUP procedure used under RSX–11M and RSX–11M–PLUS will dismount the network during the dismount phase. If you wish to clear the network software during SHUTUP, include the following commands in LB:[1,2]SHUTUP.CMD:

```
NCP SET EXECUTOR STATE OFF

.WAIT NETACP

NCP CLEAR SYSTEM
```

The .WAIT directive must be used because the SET EXECUTOR STATE OFF command completes asynchronously.

3.2 Starting Up and Shutting Down a Local DECnet-RSX/PSI Node

The following two sections contain the procedures for starting up and shutting down a DECnet-RSX/PSI or RSX-11 PSI node.

3.2.1 Starting Up a Local DECnet-RSX/PSI Node

The start-up procedure for a local node loads the Communications Executive (CEX) into memory and activates the DECnet-RSX software. RSX-11 PSI is automatically loaded, but you still have to start it up.

To load the DECnet software into memory, use one of the following methods:

- The NETINS.CMD file for newly generated systems
- NCP commands for previously installed software
- VNP commands to reboot from the system image file

3.2.1.1 Using the NETINS.CMD File

The NETINS.CMD file is a product of NETGEN and is described in detail in the *DECnet-RSX Network Generation and Installation Guide*. After running NETGEN, you must use NETINS.CMD to start up a newly generated DECnet-RSX/PSI system for the first time. To load a DECnet-RSX/PSI system using this file, enter

> @[*grp*,1]NETINS

where *grp* is the network UIC group code selected at NETGEN.

3.2.1.2 Using NCP Commands

Use the following NCP commands to load and start up a DECnet-RSX/PSI system when the network software has been installed (by a previous run of NETINS.CMD, for example). Any lines and circuits that are in the ON state in the permanent database are loaded and enabled as a result of these three commands.

```
NCP> SET SYSTEM [RET]
NCP> SET EXECUTOR STATE ON [RET]
NCP> SET MODULE X25-SERVER STATE ON [RET]
```

3.2.1.3 Using VNP Commands

Use the following VNP commands to incorporate the DECnet-RSX/PSI software components into the RSX–11M or RSX–11S system image file. You must perform an RSX SAV on the system image file before you run VNP. Once you have run VNP, DECnet-RSX/PSI will start up automatically the next time you boot the RSX operating system.

```
> VNP [RET]
Enter filename: RSX11S.SYS [RET]
VNP> SET SYSTEM [RET]
VNP> SET EXECUTOR STATE ON [RET]
VNP> SET MODULE X25-SERVER STATE ON [RET]
```

NOTES

1. On RSX–11M/M–PLUS systems, you must have installed NTINIT using VMR before you can execute a VNP SET SYSTEM command.
2. On RSX–11M/M–PLUS systems, you cannot use the VNP SET EXECUTOR STATE ON command; this function must be performed at start-up time using the NCP SET EXECUTOR STATE ON command.

3.2.2 Loading and Turning On an RSX–11 PSI Line in a Running System

After you have started up the system, you can still load and turn on a specific RSX–11 PSI line or all RSX–11 PSI lines known to the system by issuing NCP commands using one of the following formats:

SET LINE *line-id* ALL STATE ON

SET KNOWN LINES ALL STATE ON

All lines to be loaded must be in the CLEARED state. The ALL parameter specifies that the line(s) are to be loaded with the default parameter values. If you wish to change the defaults, you must specify the appropriate SET LINE command parameters See Section 2.4.3 for more information.

To cause a new line or all known lines to be loaded and turned on the next time the system is started up, use the CFE DEFINE versions of the preceding commands to set up the line(s) in the permanent database. The line(s) will be automatically loaded and turned on the next time you issue the SET SYSTEM and SET MODULE X25-SERVER STATE ON commands.

3.2.3 Shutting Down RSX-11 PSI

You can shut down selected RSX-11 PSI components or the whole RSX-11 PSI module, as described in the following sections.

3.2.3.1 Shutting Down RSX-11 PSI Components

To shut down one or all RSX-11 PSI lines, issue an NCP command using one of the following formats:

SET LINE *line-id* STATE OFF

SET KNOWN LINES STATE OFF

After shutting down a given line, you can cause the associated line to be unloaded from the system by issuing an NCP command using one of the following formats:

CLEAR LINE *line-id* ALL

CLEAR KNOWN LINES ALL

Before using the line again you must reload the line using the SET LINE command.

To shut down one or all DTEs, issue an NCP command using one of the following formats:

SET MODULE X25-PROTOCOL DTE *dte-address* STATE SHUT

SET MODULE X25-PROTOCOL KNOWN DTES STATE SHUT

The system will eventually return an event message saying that the specified DTE(s) have been shut down.

3.2.3.2 Shutting Down an RSX–11 PSI Module

Use the NCP SET MODULE X25-SERVER STATE command to shut down RSX–11 PSI immediately or in an orderly fashion, depending upon the state you specify.

State	Result
SHUT	**Orderly shutdown.** Prevents any new virtual circuit connections to the module, but does not cut off current connections. After all virtual circuits have disconnected, the system returns an event message stating that the module state is OFF.
OFF	**Immediate shutdown.** Immediately terminates all virtual circuit connections without allowing circuits to disconnect in an orderly manner.

When RSX–11 PSI operation has ceased, you can then shut down any DECnet-RSX operation on the local node. See Section 3.1.3.2.

3.3 Monitoring DECnet-RSX/PSI Nodes

DECnet provides several means of monitoring network activity once you have brought up the local DECnet system.

- You can use CFE, NCP, or VNP commands to display information about network components.
- You can use NCP to request the Event Logger (EVL) to log dynamic network events. If you log the events to an event file, you can use the Event File Interpreter (EVF) to generate a formatted event report.
- You can use the Network Display Program (NTD) to display reachable nodes or a specific node's resources.

This section provides an overview of each of these facilities.

3.3.1 NCP, VNP, and CFE Display Commands

NCP, VNP, and CFE provide commands that enable you to display information about network components on your terminal.

- The NCP SHOW command displays component information from the volatile database. For example, if a line failure or a change of state on a line or circuit occurs on a routing node, you can issue an NCP SHOW ACTIVE NODES command to determine which known nodes are now reachable. Depending on the component specified, you can select from the following types of display:

  CHARACTERISTICS displays static information about the component, such as the local node identification and relevant routing parameters for the EXECUTOR component, or the names and numbers of known network objects.

  COUNTERS provides counter information for circuits, modules, lines, nodes, and the system. See Section 2.7 for a discussion of counters.

  EVENTS (SHOW LOGGING only) displays information about events currently being logged.

  STATUS shows dynamic information that usually reflects network activity for the running network. Depending on the component, this can include the local node and its operational state, reachable and unreachable nodes, the local DTE and its operational state, and circuits and lines and their operational states.

  SUMMARY (default) includes only the most useful information derived from both static and dynamic sources. This is usually an abbreviated list of information provided for both the CHARACTERISTICS and STATUS display types.

- The VNP SHOW command displays component information from the system image file. Depending on the component, you can choose a CHARACTERISTICS, EVENTS, or SUMMARY (default) display type.

- The CFE LIST command automatically displays static information for the specified component. See the description of CHARACTERISTICS.

3.3.1.1 Network Components

The components for which you can display information are:

- Areas
- Circuits
- Lines
- Logging
- Modules
- Nodes, including the executor node and alias nodes
- Objects
- Processes
- System
- Trace

In most cases you can display information for one particular component, such as for LINE DMC-0 or for all components of the specified type that fall within one of several group classifications. The group classifications that you can specify are:

- KNOWN—Displays information for all instances of a component available to the local node; for example,

  ```
  NCP> SHOW KNOWN LINES CHARACTERISTICS RET
  ```

- SIGNIFICANT—Shows information about all instances of a known component for which information is available; for example,

  ```
  NCP> SHOW SIGNIFICANT NODES RET
  ```

- ACTIVE—Displays information for all active components: those components whose state is other than OFF or CLEARED; for example,

  ```
  NCP> SHOW ACTIVE LINES CHARACTERISTICS RET
  ```

- ALL (valid only for NCP/VNP SHOW ALIAS and SHOW MODULE X25-ACCESS commands)—Displays information for all aliases or logical destination names regardless of scope; for example,

  ```
  NCP> SHOW ALL ALIASES RET
  ```

See SHOW command descriptions in the DECnet-RSX Guide to Network Management Utilities to determine which classifications apply to each component.

3.3.1.2 Copying NCP Display Information to a File

All NCP display commands optionally allow you to direct the display information to a user-specified output file instead of displaying it on your terminal. If the specified file already exists, NCP appends the display information to the file.

Example:

```
NCP> SHOW KNOWN LOGGING TO NET.LOG [RET]
```

This example creates the file NET.LOG, under the current UIC, which will contain current summary information for all known logging on the running network.

3.3.1.3 NCP SHOW Command Examples

The following examples illustrate various NCP SHOW command displays.

Example 1:

```
NCP> SHOW EXECUTOR CHARACTERISTICS [RET]

Node characteristics as of 13-SEP-86 16:29:30

Executor node = 4.19 (BURGER)
    Identification = DECnet-RSX, Management version = 4.0.0
    Host = 4.19 (BURGER), Loop count = 1, Loop length = 40
    Loop with = Mixed, Incoming timer = 15, Outgoing timer = 10
    NSP version = 4.0.0
    Maximum links = 15, Delay factor = 32, Delay weight = 2
    Routing version = 2.0.0, Type = Nonrouting IV, Maximum circuits = 1
    Segment buffer size = 576, Verification state = On
```

Example 2:

```
NCP> SHOW KNOWN LINES [RET]

Known lines summary as of 13-SEP-86 16:32:34

line            State

DMC-0           On
DMC-1           Cleared
PCL-0           On
UNA-0           On
UNA-1           Cleared
```

Example 3:

```
NCP> SHOW LINE UNA-0 CHARACTERISTICS [RET]

Line characteristics as of 13-SEP-86 16:38:06

Line = UNA-0

    Controller = Normal
    Protocol = Ethernet
    Hardware address = AA-00-03-00-01-13
    Controller CSR = 174510, Vector = 120, Priority = 5
```

Example 4:

```
NCP> SHOW LINE PCL-0 COUNTERS [RET]

Line = PCL-0
      18820  Seconds since last zeroed
          0  Attempts to become master
          0  Process errors
          0  Device errors
```

Example 5:

```
NCP> SHOW CIRCUIT PCL-0.2 CHARACTERISTICS [RET]

Circuit characteristics as of 14-SEP-86 10:02:47

Circuit = PCL-0.2

   Cost = 2, Hello timer = 15, Listen timer = 30
   Owner = XPT
   Type = DDCMP Controller, Tributary = 8
```

Example 6:

```
NCP> SHOW KNOWN OBJECTS RET

Known objects summary as of 13-SEP-86 16:39:18
Object  Name     Copies  User     Verification

0               Single  Default  Off
15      TCL...  Single  Default  On
16      LSN$$$  5       Default  Off
17      FAL$$$  8       Login    On
18      HLD...  Single  Default  Off
19      NIC$$$  5       Default  Inspect
23      RMHACP  Single  Default  Off
25      MIR$$$  5       Default  Off
26      EVR...  Single  Default  Off
63      DTR...  Single  Default  Off
```

3.3.2 Event Logging

Event logging allows you to monitor selected network activity that might otherwise go unobserved. The events that can be logged include those occurring at the local node and at remote nodes. Refer to Section 2.6 for a general overview of event logging. The Event File Interpreter program (EVF), which can be used to generate formatted event reports from an event file, is described in the *DECnet-RSX Guide to Network Management Utilities* .

The following network management commands control event logging. See the *DECnet-RSX Guide to Network Management Utilities* for command details:

CFE Commands	NCP/VNP Commands
DEFINE LOGGING	SET LOGGING
PURGE LOGGING	CLEAR LOGGING
LIST LOGGING	SHOW LOGGING

A simple example of how you can use event-logging commands to monitor local and remote nodes in your network is shown in the following sample. This sequence of NCP commands sets up event logging for a local node called ALB and for two remote nodes called PHL and BOS. In response to the first command, ALB's system console will display all known events that occur locally. In response to the next four commands, remote nodes BOS and PHL will each log all known events locally at the system console and remotely at node ALB's system console.

```
NCP> SET LOGGING CONSOLE KNOWN EVENTS STATE ON [RET]
NCP> TELL BOS SET LOGGING CONSOLE KNOWN EVENTS STATE ON [RET]
NCP> TELL BOS SET LOGGING CONSOLE SINK NODE ALB KNOWN EVENTS [RET]
NCP> TELL PHL SET LOGGING CONSOLE KNOWN EVENTS STATE ON [RET]
NCP> TELL PHL SET LOGGING CONSOLE SINK NODE ALB KNOWN EVENTS [RET]
```

Appendix D describes the event message format and lists the text of all DECnet-RSX event messages.

3.3.3 The Network Display Program (NTD)

You can run the Network Display Program (NTD) on the local node or on any RSX or IAS node in the network. NTD can produce three types of real-time displays:

- A list of all reachable areas in the network provided the node is an area router
- A list of all reachable nodes in the network provided the node is a routing node
- A resource display for a specified node provided the node supports NTDEMO, the task that collects the information that is displayed

You can display this information on a local VT52, VT100, or VT200-series terminal, where it is continuously updated, or you can obtain a snapshot printout of the display information on a hard-copy terminal. The DECnet-RSX Guide to Network Management Utilities describes NTD in detail.

4
Testing the Network

DECnet-RSX and RSX–11 PSI provide several kinds of loopback tests to help you determine if the network is operating properly. Loopback tests let you exercise network software and hardware by sending data through various network components and then returning that data to its source for data comparison. After you have started DECnet-RSX or RSX–11 PSI software, you can perform these tests using NCP commands, user-written programs, or DECnet-supplied utilities. This chapter describes these test facilities and provides a practical approach to their use.

NOTE

The tests described in this chapter are general tests that can be used whenever a problem in the network is suspected. DECnet-RSX and RSX–11 PSI also provide a special set of test procedures that should be used whenever a new node is first installed. These tests are described in the *DECnet-RSX Network Generation and Installation Guide*.

DECnet-RSX tests fall into two categories:

- **Node level loopback tests** evaluate the operation of logical links, routing, and other network-related software. See Section 4.1.
- **Circuit level loopback tests** evaluate the operation of circuits. See Section 4.2.

You should use the node level tests first, then, if necessary, use the circuit level tests.

RSX–11 PSI provides various ways to analyze software and hardware operations and to diagnose problems in RSX–11 PSI operations:

- **Line level loopback tests**, described in Section 4.3, evaluate the operation of the X.25 physical lines and communications hardware.
- **The X.25 trace interpreter task**, described in Section 4.4, records the flow of packets and frames for future analysis. The trace analyzer analyzes the trace data.
- **The KMX/KMY/KMV microcode dump analyzer**, described in Section 4.5, allows you to dump the KMX/KMY/KMV (KMS-11) microcode to a specified file for analysis.

4.1 Node Level Tests

Node level loopback tests test the logical link capabilities of a node by exchanging test data between DECnet tasks in two different nodes or between DECnet tasks in the same node. There are four types of node level tests:

- **Local-to-remote loopback tests.** Verify local and remote node network operations. See Section 4.1.1.
- **Local-to-remote loopback tests using a specific circuit.** Verify local and adjacent node network operations over a specific circuit. See Section 4.1.2.
- **Local-to-local loopback tests using a specific circuit.** Verify local node network operations on a specific circuit. See Section 4.1.2.
- **Local-to-local loopback tests.** Verify local node network operations. See Section 4.1.3.

The four types of node level loopback tests allow you to test various layers of DECnet-RSX software. To test these layers in a methodical manner, perform the following series of operations:

1. First, test the operation of both the local node and a remote node by using the local-to-remote loopback test described in Section 4.1.1.
2. If the first test fails, set up a loop node name, as described in Section 4.1.2, specifying the circuit that is in the path to the remote node. Following the procedure described in Section 4.1.2.1, use this loop node to test the circuit to the adjacent node in the path. If necessary, this process can be repeated on all nodes in the path to the desired remote node.

3. If the second test fails, use a loopback connector, modem, or set the device to controller loopback mode and the same loop node name to test the circuit within the local node as described in Section 4.1.2.2.

4. If the third test fails, test the operation of your local node by using the Local-to-local test described in Section 4.1.3.

To perform these loopback tests, use the NCP LOOP NODE command. This NCP operation uses the two cooperating tasks, loopback tester (LOOPER) and loopback mirror (MIR),to perform the loopback tests. When you issue this command, you have the option of controlling:

- The type of binary information the test is performed with: WITH MIXED, ONES, or ZEROES.
- The COUNT of blocks of information, which ranges from 1 to 65,535.
- The LENGTH of each block to be looped, which ranges from 1 to 65,535 bytes. It is recommended that you use a maximum block length of 4096 bytes to reduce the system load.

Refer to the NCP chapter in the DECnet-RSX Guide to Network Management Utilities for the complete syntax of the LOOP NODE command.

If the test completes successfully, NCP prompts you for the next command. If a looped message returns with an error, the test stops and NCP prints a message that indicates a test failure, specifies the reason for the failure, and provides a count of the messages that were not returned. For a summary of NCP error messages, refer to the DECnet-RSX Guide to Network Management Utilities.

In the following example, the test attempts to send 10 messages, each 50 bytes long. The first two messages complete successfully, and an error occurs on the third.

```
NCP> LOOP NODE BOSTON COUNT 10 RET

NCP> -- LOOP FAILED, cause-text

Unlooped count = 8
```

The *cause-text* string gives additional information about the cause of the test failure.

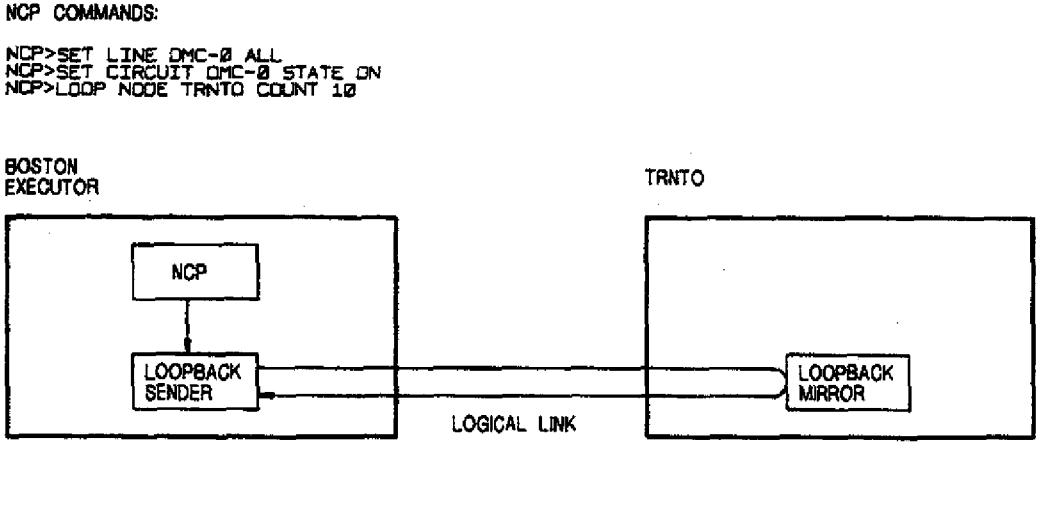
4.1.1 Local-to-Remote Loopback Test

This test verifies the operation of all the levels of network software in the two nodes under test and up to the Routing layer in any intermediate nodes. When using this command, you must identify the node to which you want to loop test messages. This node must be reachable over the circuits that are in the ON state. Figure 4-1 illustrates a remote loopback test.

For this test, you first load the selected line and turn the circuit to the ON state to allow for logical link activity. Then use the LOOP NODE command to initiate the loopback test. For example, the following set of commands tests both local and remote DECnet software on nodes BOSTON and TRNTO.

```
NCP> SET LINE DMC-0 ALL [RET]
NCP> SET CIRCUIT DMC-0 STATE ON [RET]
NCP> LOOP NODE TRNTO COUNT 10 [RET]
```

Figure 4–1: Local-to-Remote Loopback Test



A failure in this test indicates that a problem exists in the network software in the local node, the remote node, or any of the intermediate nodes. To further isolate the problem you should perform the other node level loopback tests described in the next sections.

4.1.2 Loopback Tests Using a Loop Node to Specify the Circuit

NOTE

The two tests described in this section cannot be performed with Ethernet circuits because DECnet-RSX does not allow Ethernet circuits to have a loop node associated with them. However, you can use the circuit level loopback tests described in Section 4.2.

If the local-to-remote loopback test fails, then use the LOOP NODE command with a loop node name to test a local logical link path over a specific circuit. You can loop test messages over a logical link path and circuit to the adjacent node on the circuit or loop the messages totally within the local node and its hardware by setting the line to LOOPBACK state. Use the SET NODE command with the CIRCUIT parameter to establish a loop node name. For example, the following command establishes circuit DMC-0 as the circuit over which loop testing will take place:

```
NCP> SET NODE TESTER CIRCUIT DMC-0 [RET]
```

Setting up a loop node name is necessary because, under normal operation, the DECnet Routing layer decides what path to use when routing information. The loop node name overrides the routing function so that information can be routed over a specific circuit. When you make a logical link connection request to a loop node name, all subsequent logical link traffic is directed over the circuit associated with the loop node name. The destination of the traffic is the adjacent node on the specified circuit.

A loop node name specified with the SET NODE CIRCUIT command may be used for any network traffic: for example, NFT requests or application program traffic. The loopback node name appears as a valid node name in the network.

Note that you cannot assign two loop node names to the same circuit. For example, once you establish TESTER as the loop node name for circuit DMC-0, you must issue a CLEAR NODE TESTER CIRCUIT command before assigning another loop node name to DMC-0.

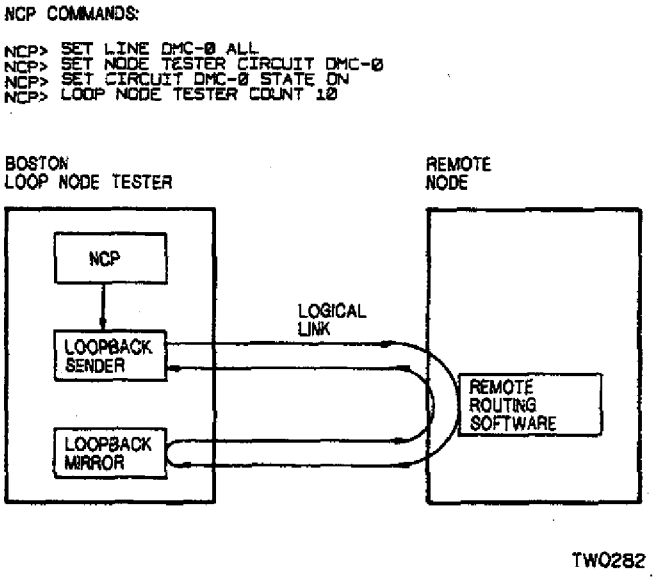
To remove the association of the loop node name with a circuit, use the CLEAR NODE CIRCUIT or CLEAR NODE ALL command, as in the following command:

```
NCP> CLEAR NODE TESTER CIRCUIT [RET]
```

4.1.2.1 Local-to-Remote Loop Node Testing

Figure 4-2 illustrates a local-to-remote loopback test using a loop node name. This test verifies both the local and adjacent node's Routing layer and Data Link layer software operation. The test messages are looped over the circuit associated with the specified loop node. Because the test actually tests the operation of the Routing layer on the adjacent node, the message may not come back on the circuit over which it was sent.

Figure 4–2: Local-to-Remote Loopback Test Using a Loop Node Name



For this test, you first load the line and set a loop node name for the circuit to the adjacent node. Next, turn on the circuit. Finally, issue the LOOP NODE command using the loop node name, as shown in the following example:

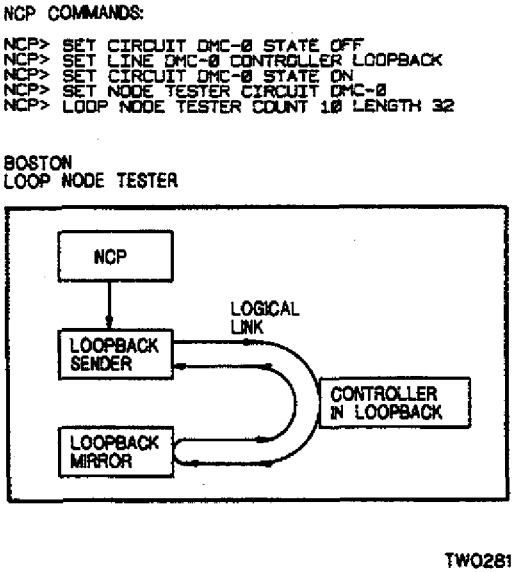
```
NCP> SET LINE DMC-0 ALL [RET]
NCP> SET NODE TESTER CIRCUIT DMC-0 [RET]
NCP> SET CIRCUIT DMC-0 STATE ON [RET]
NCP> LOOP NODE TESTER COUNT 10 [RET]
```

A failure in this test indicates a problem exists with the network software in the local node or adjacent node. To isolate the problem further, perform the other node level tests described in the next sections.

4.1.2.2 Local-to-Local Loop Node Testing

If the local-to-remote loop node test fails, perform a local-to-local loop node test. Figure 4-3 illustrates a local-to-local loopback test using a loop node name.

Figure 4–3: Local-to-Local Loopback Test Using a Loop Node Name



This test verifies the local Routing layer software exclusively. To test a logical link path over a specified circuit on the local node, specify a loop node name and set up the device in a LOOPBACK state by setting the line's controller to loopback mode, using a loopback connector, or using a modem in loopback mode.

For this test, you first turn off the circuit and set the device to a LOOPBACK state or set up the loopback hardware as described in Section 4.2.3.1. Next, turn on the circuit. Finally, set a loop node name for the circuit and issue the LOOP NODE command using the loop node name as shown in the following example:

```
>NCP> SET CIRCUIT DMC-0 STATE OFF [RET]
NCP> SET LINE DMC-0 CONTROLLER LOOPBACK [RET]
NCP> SET CIRCUIT DMC-0 STATE ON [RET]
NCP> SET NODE TESTER CIRCUIT DMC-0 [RET]
NCP> LOOP NODE TESTER COUNT 10 LENGTH 32 [RET]
```

Because the device is set up to a LOOPBACK state, the test messages are looped over the circuit and back to the local node.

A failure in this test indicates that a problem exists with the network software in the local node or in the device hardware. If this test fails, you should perform a local-to-local loopback test as described in the next section to test the local DECnet software.

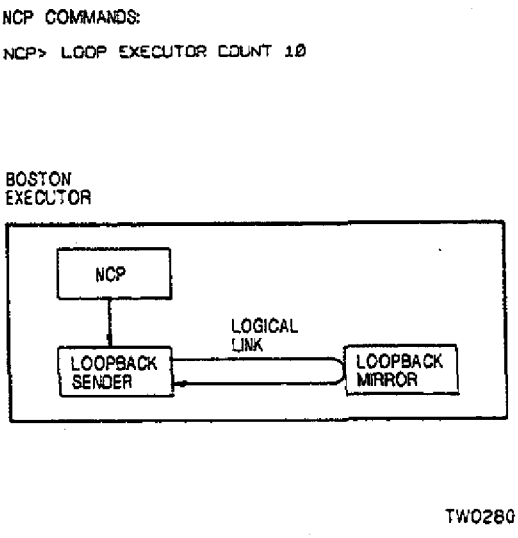
NOTE

Because of restrictions in the operation of the DMC controller, you must use a block length of fewer than 50 bytes for controller loopback tests.

4.1.3 Local-to-Local Loopback Test

If both loopback tests using a loop node fail, perform a local-to-local test to verify the operation of the local node software. Figure 4-4 illustrates a local loopback test.

Figure 4–4: Local Loopback Test



For this test, you simply issue the LOOP EXECUTOR command as shown in the following example:

```
NCP> LOOP EXECUTOR COUNT 10 RET
```

This test evaluates the local DECnet software using an internal logical link path; no physical device is used in this type of test. The test verifies the operation of the local Network Management layer, User layer, Network Application layer, Session Control layer, End Communications layer, and part of the Routing layer. If this test succeeds and the other node level tests fail, then try the circuit level tests to determine if the hardware is at fault. If this test fails, there is a problem with the local node software.

4.2 Circuit Level Tests

Circuit level loopback tests test a DECnet circuit by looping test data between DECnet tasks in two different nodes or between DECnet tasks in the local node. These tests use a low-level data link interface rather than the logical links used by the node level tests. There are three types of circuit level tests:

- **Software loopback tests**, described in Section 4.2.1, loop the data through the adjacent node on the circuit.
- **Controller loopback tests**, described in Section 4.2.2, loop the data through the device on the circuit in loopback mode.
- **Circuit loopback tests**, described in Section 4.2.3, loop the data through a modem, or loopback connector, attached to the circuit device.

To test various components of a circuit you can perform a series of operations, which are described in the following list.

1. In the first test, perform a software loopback test to an adjacent node to determine whether the circuit is operational up to the adjacent node's circuit unit and controller. This type of test can be repeated all the way along a path between two nodes in order to isolate the problem circuit.
2. If the first test fails and the device supports the loopback mode of operation, set the controller to loopback mode and use a controller loopback test to determine whether the controller works.
3. If the second test succeeds or if the device does not support the loopback mode of operation, attach a modem or a loopback connector to the controller and use a circuit loopback test to determine whether the unit and the associated cable are functional.

To perform these loopback tests, use the NCP LOOP CIRCUIT command. This NCP operation uses the cooperating processes LOOPER and LIN to perform the loopback tests. When you issue this command, you have the option of controlling:

- The type of binary information the test is performed with: WITH MIXED, ONES, or ZEROES.
- The COUNT of blocks of information, which ranges from 1 to 65,535.
- The LENGTH of each block to be looped, which ranges from 1 to 65,535 bytes. It is recommended that you use a maximum block length of 4096 bytes.

For the complete syntax of the LOOP CIRCUIT command, refer to the NCP chapter in the *DECnet-RSX Guide to Network Management Utilities* .

If your message returns with an error, the test stops, and NCP prints a message indicating a test failure, the reason for the failure, and a count of the messages that were not returned—for example,

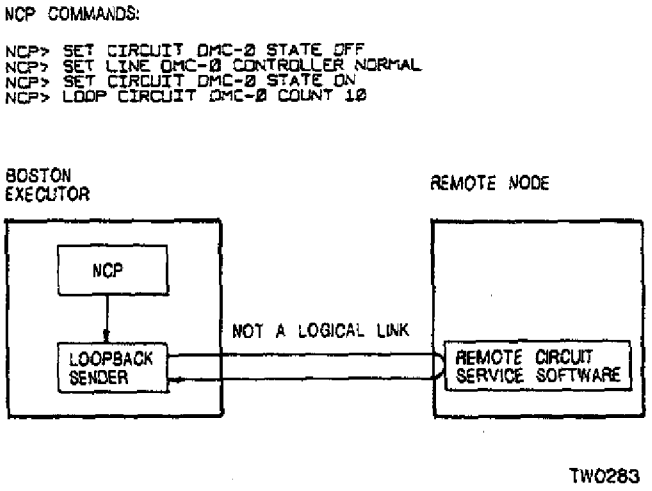
```
NCP> SET CIRCUIT DUP-0 STATE SERVICE [RET]
NCP> LOOP CIRCUIT DUP-0 COUNT 10 [RET]

NCP -- Loop failed, line protocol error
Unlooped count = 8
```

In this test, 10 messages were sent. The first two messages were sent successfully, and an error occurred on the third. When the error occurred, the test halted.

4.2.1 Software Loopback Test

This test verifies that the circuit is operational up to the unit and controller on the adjacent node. This type of test uses DECnet-RSX software to loop data through the circuit to circuit service software in the adjacent node and back to the local node. For Ethernet circuits, you can specify an optional parameter that controls the adjacent node that will be used for testing. See Section 4.2.1.1 for Ethernet testing. Figure 4-5 illustrates a software loopback test.

Figure 4–5: Circuit Level Loopback Test



In the first step of this test, you turn the circuit off. Next, you set the controller to its normal operational mode and put the circuit in the ON state. Finally, you issue the LOOP CIRCUIT command. This sequence is shown in the following example:

```
NCP> SET CIRCUIT DMC-0 STATE OFF [RET]
NCP> SET LINE DMC-0 CONTROLLER NORMAL [RET]
NCP> SET CIRCUIT DMC-0 STATE ON [RET]
NCP> LOOP CIRCUIT DMC-0 COUNT 10 [RET]
```

This set of commands tests the circuit DMC-0 up to the adjacent node. If this test fails and the device supports the loopback mode of operation, try a controller loopback test to verify that the controller is functional.

4.2.1.1 Software Loopback Testing over Ethernet Devices

There are two ways of performing the software loopback test on the Ethernet:

- Loop to any random node on the Ethernet
- Loop to a specific node on the Ethernet

In order to be tested, an Ethernet circuit must be in the ON state and the SERVICE parameter must be set to ENABLED. Note that, by default, the SERVICE parameter is set to ENABLED for Ethernet circuits. In the following example, the command identifies the circuit device UNA and the controller number 0 for an Ethernet circuit.

```
NCP> SET CIRCUIT UNA-0 STATE ON SERVICE ENABLED [RET]
```

Random Node Loopback Testing: You can use the following command to send a test message to all nodes on the Ethernet by way of a multicast message. If the COUNT parameter is greater than 1, the first node to respond to the multicast message will be used to loop the remaining test messages. For example,

```
NCP> LOOP CIRCUIT UNA-0 COUNT 50 [RET]
```

A return message is displayed that indicates the node on the Ethernet that responded to the loop:

```
NCP -- Loop Succeeded
       Physical Address=AA-00-04-00-23-04, Node = 35 (NODEX)
```

Specific Node Loopback Testing: You can specify the physical address or the DECnet node name/address for a specific remote node on the circuit that you want to test.

NOTE

Nodes on Ethernet circuits are identified by unique Ethernet addresses. If the node is running DECnet, this physical address is the address that DECnet has created using the DECnet node address. If the node is not running DECnet, the physical address is the hardware address of the node. For more information on Ethernet physical addresses and hardware addresses, see Section 2.1.5.

To perform the test using a specific remote node, specify the PHYSICAL ADDRESS parameter along with its value. The following command loops messages through the local device UNA-0 to the remote node having physical address AA-00-03-00-FF-08:

```
NCP> LOOP CIRCUIT UNA-0 PHYSICAL ADDRESS AA-00-03-00-FF-08 [RET]
```

4.2.1.2 Ethernet Loopback Assistance

DECnet supports the use of an **assistant node** to aid you in interrogating a remote node. To use this assistant feature, you specify either the ASSISTANT PHYSICAL ADDRESS parameter or the ASSISTANT NODE parameter as an additional parameter to the LOOP CIRCUIT command.

You can specify the assistant in three distinct ways using the HELP parameter.

- HELP RECEIVE—The assistant aids in receiving loop messages from a remote node.
- HELP TRANSMIT—The assistant aids in transmitting loop messages to a remote node.
- HELP FULL—The assistant aids in both transmitting messages to and receiving messages from a remote node.

There are various reasons why you might choose one form of assistance over another. For example, it is possible that the target node to which you wish to transmit a message is not receiving messages from your node. In this case, you can request assistance in transmitting the message to the target node. Similarly, your node may be able to transmit messages to the target node, but not be able to receive messages from it. In such a case you can send a message directly to the target node and request the assistant's aid in receiving a message from the target node. When you encounter difficulties in both sending and receiving messages, you can request an assistant's aid in both transmitting and receiving messages from the target node. Note that in order for the assistant node to be of any use the assistant must be between the local node and the remote node. The three types of assistance are shown in Figures 4–6 to 4–8.

The following commands illustrate the use of the ASSISTANT PHYSICAL ADDRESS and ASSISTANT NODE parameters. The commands are shown here in continuation format.

```
NCP> LOOP CIRCUIT UNA-0 PHYSICAL ADDRESS AA-00-04-00-18-04 -[RET]
 NCP> ASSISTANT PHYSICAL ADDRESS AA-00-04-00-15-04 [RET]

NCP> LOOP CIRCUIT UNA-0 NODE LOON ASSISTANT NODE THRUSH -[RET]
 NCP> HELP TRANSMIT [RET]
```

In the first command, you are requesting the node described by the Ethernet physical address AA-00-04-00-15-04 to assist you in testing the node described by the Ethernet physical address AA-00-04-00-18-04. In the second command, you are requesting node THRUSH to assist in testing node LOON by transmitting the loopback data to node LOON.

Figure 4–6: Loopback Test Using an Assistant Giving Full Assistance

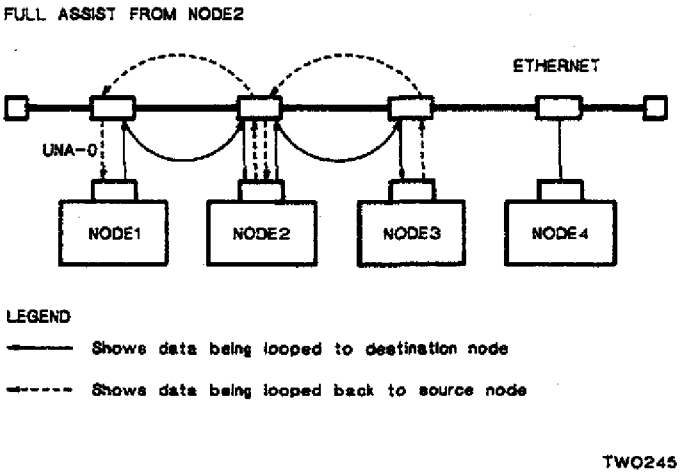


Figure 4–7: Loopback Test Using an Assistant Giving Transmit Assistance

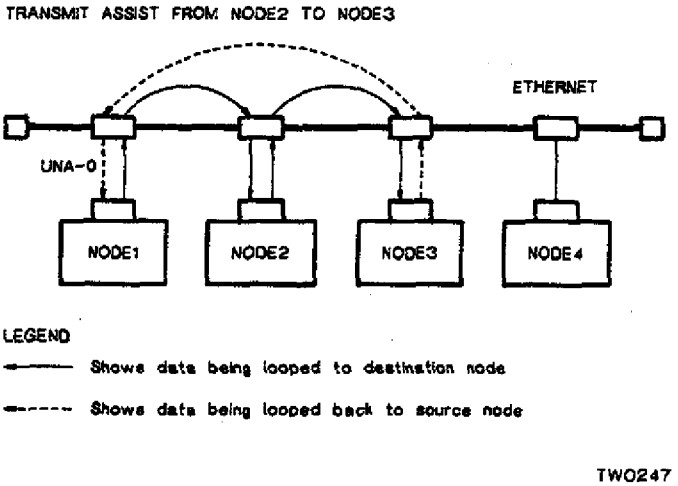
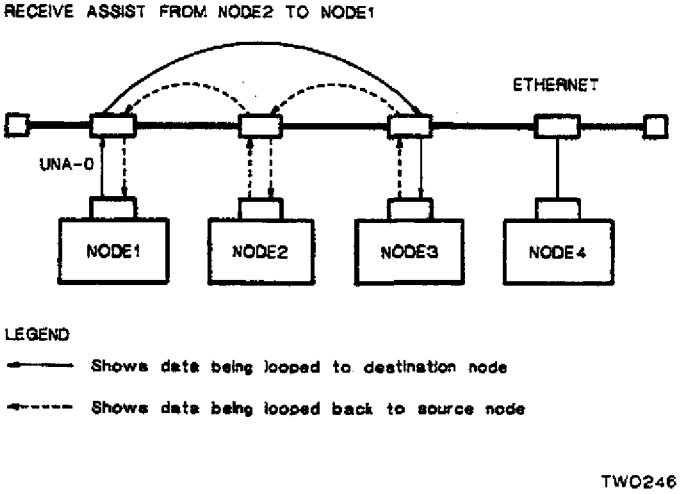


Figure 4–8: Loopback Test Using an Assistant Giving Receive Assistance

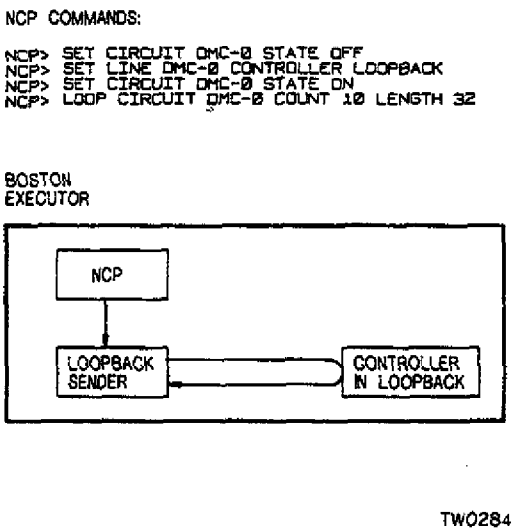


If you specify either the ASSISTANT PHYSICAL ADDRESS or ASSISTANT NODE parameter and you do not specify the HELP parameter, you will receive FULL assistance by default. In the first example above, the ASSISTANT PHYSICAL ADDRESS was specified without the HELP parameter; therefore, FULL assistance was given.

4.2.2 Controller Loopback Test

This type of test can only be done on devices which support the loopback mode of operation: DMC, DMR, DMP, DHU, DHV, UNA, and QNA. This type of test verifies whether or not the circuit up to the controller and the controller itself are functional. Figure 4–9 illustrates a controller loopback test.

Figure 4–9: Controller Loopback Testing



For this test, you first turn the circuit off. Next, you set the controller to loopback mode and put the circuit in the ON state. Finally, you issue the LOOP CIRCUIT command. For example:

```
NCP> SET CIRCUIT DMC-0 STATE OFF [RET]
NCP> SET LINE DMC-0 CONTROLLER LOOPBACK [RET]
NCP> SET CIRCUIT DMC-0 STATE ON [RET]
NCP> LOOP CIRCUIT DMC-0 COUNT 10 LENGTH 32 [RET]
```

This set of commands tests the circuit up to the controller for physical line DMC-0 connected to the local node by circuit DMC-0.

If this test succeeds and the software loopback test fails, perform a circuit loopback test to verify the operation of the device and device cable.

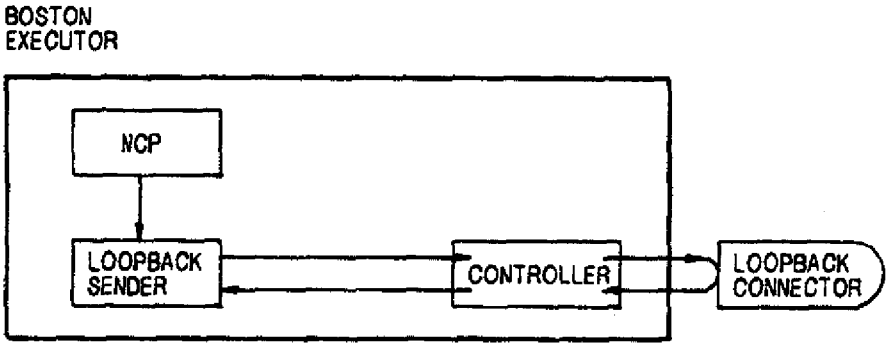
NOTE

Because of restrictions in the operation of the DMC controller, you must use a block length of fewer than 50 bytes for controller loopback tests.

4.2.3 Circuit Loopback Tests

This type of test verifies whether or not the device and cable are functional. Figure 4-10 illustrates a circuit loopback test with a loopback connector. See Figure 4-11 for other hardware loopback arrangements.

Figure 4–10: Circuit Loopback Testing Using a Loopback Connector



TWO285

For this test, put the circuit in the SERVICE or the ON state. The test can be started from either the ON or the SERVICE state. However, the SERVICE state is the preferred state for this test, since it allows testing that does not affect the network as a whole. Then use the LOOP CIRCUIT command. For example,

```
NCP> SET LINE DUP-0 DUPLEX FULL [RET]
NCP> SET CIRCUIT DUP-0 STATE SERVICE [RET]
NCP> LOOP CIRCUIT DUP-0 COUNT 10 [RET]
```

This set of commands tests the circuit DUP-0 up to the loopback connector.

4.2.3.1 Hardware Arrangements for Circuit Loopback

The hardware arrangement for circuit loopback depends on your installation's interface equipment and the hardware components you wish to include in the test.

- **Loopback connector.** The loopback connector is installed on the cable between the line device and the modem, as shown in Figure 4-11(a). Table 4-1 lists the currently supported DECnet devices and their loopback connectors. This configuration tests the line device and the cable. The connector returns digital data to the line device. You must condition the line device for full duplex mode by using the SET LINE DUPLEX FULL command. Wiring diagrams for the H315 and H325 loopback connector are shown in Figures 4-12 and 4-13 respectively.

- **Connected devices.** If you have more than one similar line on your system, you can use two lines for line loopback by connecting their line devices, either directly by way of coaxial cable or over a telephone circuit by way of modems as shown in Figure 4-11(D). This loopback arrangement tests both line devices and whatever cables and modems connect them. The data is not forced back in the hardware, but received by a device at the local node instead of at a remote node. This is analogous to having two telephones in your house and using one to call the other. This is the only loopback arrangement whereby you can run the line devices in half duplex mode. Full duplex mode is also permitted.

- **Local modem loopback.** Some modems, for example 208A and 208B types, have an analog loopback (AL) button. The button directs the modem's transmitter output to its receiver input. This loopback arrangement tests the line device, the cable, and the modem, as shown in Figure 4-11(B). The modem converts data from the device to analog form, digitizes the data again, and returns it to the device. You must condition the device for full duplex mode by using the SET LINE DUPLEX FULL command. Note that local modem loopback must always operate in full duplex mode, even for 208B type modems whose telephone circuits are half duplex.

  Some modems may need additional modification in order to perform loopback tests. The breakout box shown in Figure 4-15 may be required to allow the loopback of specific modem handshaking signals that are normally set OFF when the modem is in an analog loopback state. The communications software requires that DATASET-READY be ON. Some modems, such as the Bell 201C, set DATASET-READY to OFF when in the analog loopback state.

When installing the breakout box between the modem cable and the modem, make the following patches across the DTE side of the breakout box:

- Patch DATASET-READY (pin 6) to DATATERMINAL-READY (pin 20).
- Patch REQUEST-TO-SEND (pin 4) to CLEAR-TO-SEND (pin 5).

Set switches 5 and 6 on the breakout box to OFF. Set the remaining switches to ON. The DTE side of the breakout box is the side connected to the computer, as opposed to the side connected to the modem, which is called the DCE side.

- **Remote modem loopback.** Some modems, for example 208A types, have a digital loopback (DL) button. The button directs the modem's receiver output to its transmitter input. A remote modem set in the DL state and connected to the local modem by a full duplex telephone circuit is shown in Figure 4-11(C). This loopback arrangement tests the device, the cable to the local modem, the local modem, the telephone circuit, and the remote modem. The remote modem digitizes data from the local modem, converts the data to analog form, and returns it to the local modem. The data is then returned to the device. You must condition the device for full duplex mode by using the SET LINE DUPLEX FULL command.

Table 4–1: Digital Communications Interface Specifications for Loopback Processing

Interface Name	Interface Type	External Loopback Connector	External Loopback Capability	Cables Required[1]
DHU11	EIA	H325	Yes	BC22F (Shielded), BCO5D (Unshielded)
DHV11 and DHQ11	EIA	H325	Yes	BC22F (Shielded), BC05D (Unshielded)
DL11-E	EIA	H315	Yes	Cable supplied in kit
DLV11-E	EIA	H315	Yes	BC05C
DMC11-AL	Local Sync	12-12528 coaxial test connector	Yes	BC03N

[1]The cables used depend on the type of modem in the system.

Table 4–1 (Cont.): Digital Communications Interface Specifications for Loopback Processing

Interface Name	Interface Type	External Loopback Connector	External Loopback Capability	Cables Required[1]
DMC11-AR with a DMC11-DA line unit	RS232-C/CCITT V.24	H325	Needs a clocked null modem	BC05D
DMC11-AR with DMC11-FA line units	RS232-C/CCITT V.24	H325	Needs a clocked null modem	BC05Z
DMP11-AA	RS232-C, RS423-A	H325, H3251	Yes	BC55C-10, BC05D-25
DMP11-AA	RS232-C, RS423-A	H325, H3251	Needs a clocked null modem	BC55C-10 and null modem
DMP11-AB	CCITT V.35	H3250	Yes	BC05Z-25
DMP11-AC	Local sync at 56 bps or 1M bps	Half duplex switch ON and cables removed	Yes[3]	BC55A-10 and Twinax cables
DMP11-AE	RS-422-A	H3251	Yes	BC55B-10, BC55D-33
DMR11-AA with Bell 200 modem or equiv	EIA, RS232-CCITT V.24	H325	Yes [3]	BC05D 25 pin
DMR11-AA	RS449/423	H3251	Yes[3]	BC55D 37 pin
DMR11-AB[4] with Bell	CCITT V.35/DDS	H3250	Yes[3]	BC05Z-25
500AL1/5 or equiv				
DMR11-AC	Local sync 4 selectable speeds up to 1M bps	Half duplex switch ON and cables removed	Yes[3]	BC03N, BC55M, or BC55N
DMR11-AE with Bell 303C type modems	EIA, RS 449/422	H3250	Yes	BC05Z-25
DU11-DA	EIA	H315	Needs a clocked null modem	BC02C

[1]The cables used depend on the type of modem in the system.

[3]A LOOP CIRCUIT command does not work with this interface. Use LOOP NODE.

[4]For RSX systems only.

(continued on next page)

Table 4–1 (Cont.): Digital Communications Interface Specifications for Loopback Processing

Interface Name	Interface Type	External Loopback Connector	External Loopback Capability	Cables Required[1]
DUP11-DA	EIA	H325	Needs a clocked modem	BC02C
DUV-DA (LSI-11)	EIA	H325	Needs a clocked modem	BC05C
DV11	EIA	H325	Needs a clocked modem	BC05D
DZ11-A,B,H	EIA	H325	Yes	BC05D
DZV11	EIA	H325	Yes	BC11U
PCL11	Parallel	Not required	Yes	BC20K, BC20P

[1]The cables used depend on the type of modem in the system.

4.3 X.25 Line Level Loopback Tests

To test an X.25 physical line you can perform a line level loopback test by looping data back through the modem. Line level loopback tests can be performed only on SDP (DUP-1) and SDV (DPV-11) devices.

To perform this loopback test use the NCP LOOP LINE command. This NCP operation uses the two cooperating processes LOOPER and LIN. When you issue this command you have the option of controlling:

- The type of binary information the test is performed with: WITH MIXED, ONES, or ZEROES. If this parameter is not specified, the default is MIXED ones and zeroes.
- The COUNT of the blocks of information, which ranges from 1 to 65,535. If this parameter is not specified, the default is 1.

- The LENGTH of each block to be looped, which ranges from 1 to 65,535 bytes. It is recommended that you use a maximum block size of 4096 bytes to reduce the system load. If this parameter is not specified, the default is 128 bytes.

Refer to the NCP chapter in the *DECnet-RSX Guide to Network Management Utilities* for the complete syntax of the LOOP LINE command.

Figure 4–11: Hardware Loopback Arrangements

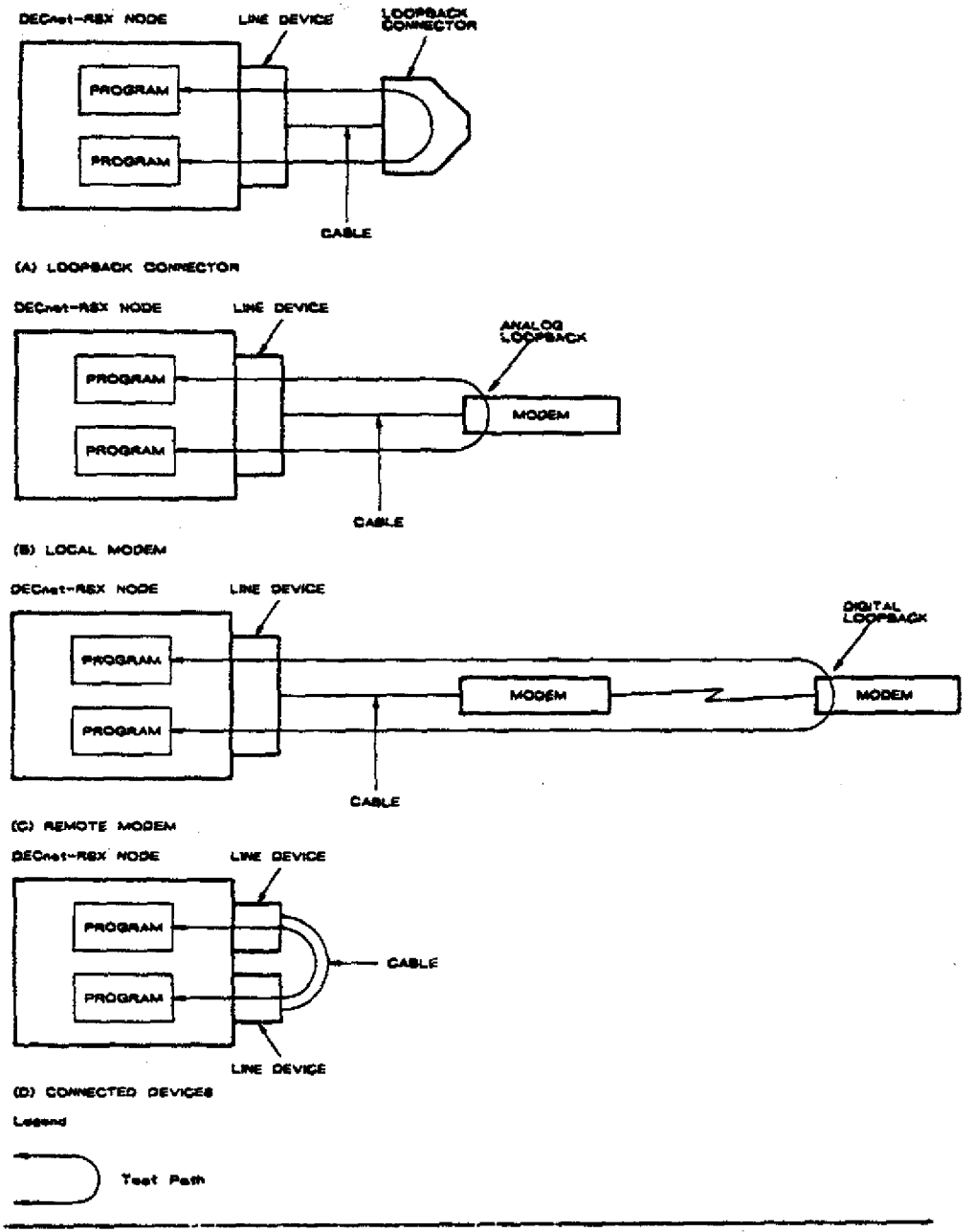
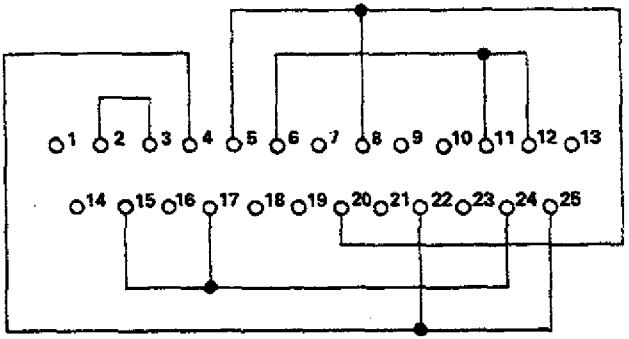


Figure 4–12: H315 Loopback Connector Wiring Diagram



Use this connector to test DQ, DU, and DL11-E interfaces.

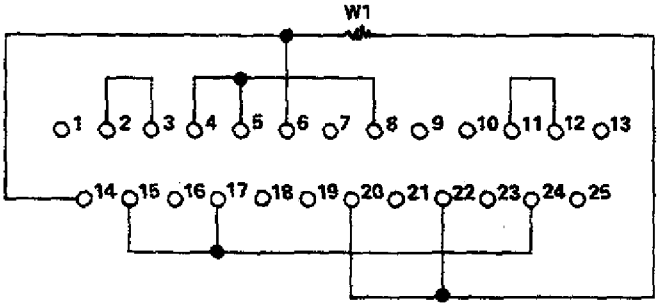
View Looking at Back of Connector

TERMINAL	SIGNAL
2	SEND DATA
3	RECEIVE DATA
4	SEND REQUEST
5	CLEAR TO SEND
6	DATA SET READY
8	CARRIER/DETECT
11	SEC'Y XMIT DATA
12	SEC'Y REC DATA
15	CLOCK XMIT
17	CLOCK REC
20	TERMINAL READY
22	RING
24	EXTERNAL CLOCK
25	BUSY

SIGNALS	
SEND REQUEST	4
BUSY	25
RING	22
SEND DATA	2
RECEIVE DATA	3
EXTERNAL CLOCK	24
CLOCK REC	17
CLOCK XMIT	15
TERMINAL READY	20
CARRIER/DETECT	8
CLEAR TO SEND	5
DATA SET READY	6
SEC'Y XMIT DATA	11
SEC'Y REC DATA	12

LKG-1114-87

Figure 4–13: H325 Loopback Connector Wiring Diagram



Use this connector to test DV, DUP, DZ, and DMC interfaces. When used with the DV11 interface, remove W1.

View Looking at Back of Connector

TERMINAL	SIGNAL
2	SEND DATA
3	RECEIVE DATA
4	SEND REQUEST
5	CLEAR TO SEND
6	DATA SET READY
8	CARRIER/DETECT
11	SEC'Y XMIT DATA
12	SEC'Y REC DATA
14	SEC XMIT DATA
15	CLOCK XMIT
17	CLOCK REC
20	TERMINAL READY
22	RING
24	EXTERNAL CLOCK

SIGNALS	
DATA SET READY	6
SEC XMIT DATA	14
	W1
TERMINAL READY	20
RING	22
SEND DATA	2
RECEIVE DATA	3
EXTERNAL CLOCK	24
CLOCK REC	17
CLOCK XMIT	15
SEND REQUEST	4
CARRIER/DETECT	8
CLEAR TO SEND	5
SEC'Y XMIT DATA	11
SEC'Y REC DATA	12

LKG-1115-87

Figure 4–14: EIA Asynchronous Null Modem Wiring Diagram

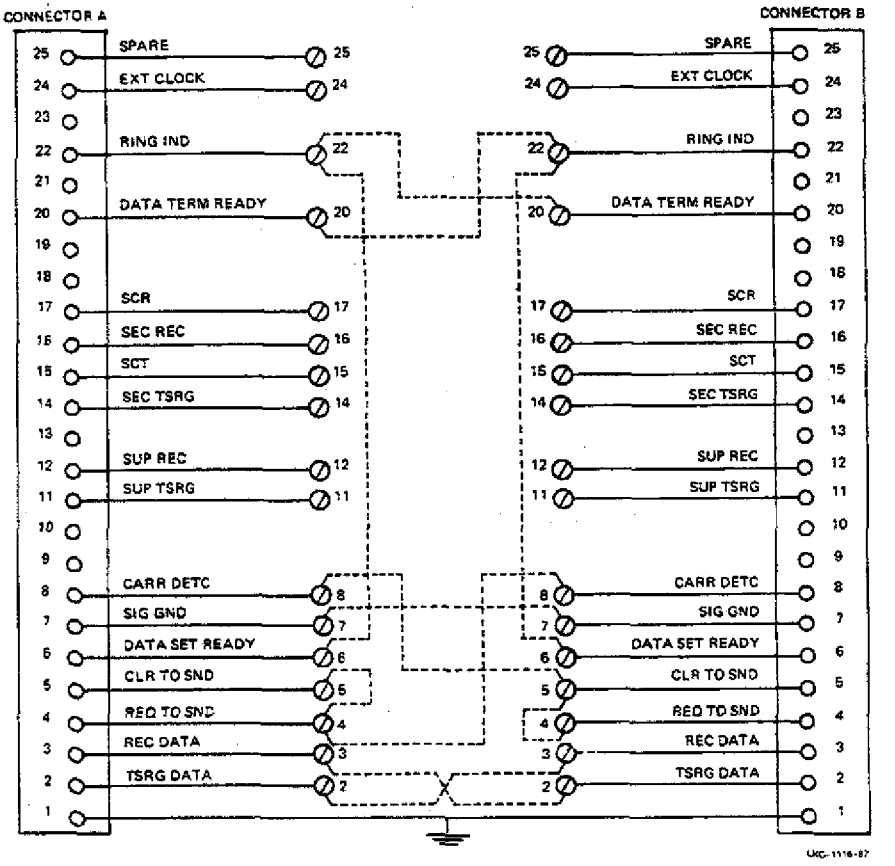


Figure 4–15: Typical Breakout Box

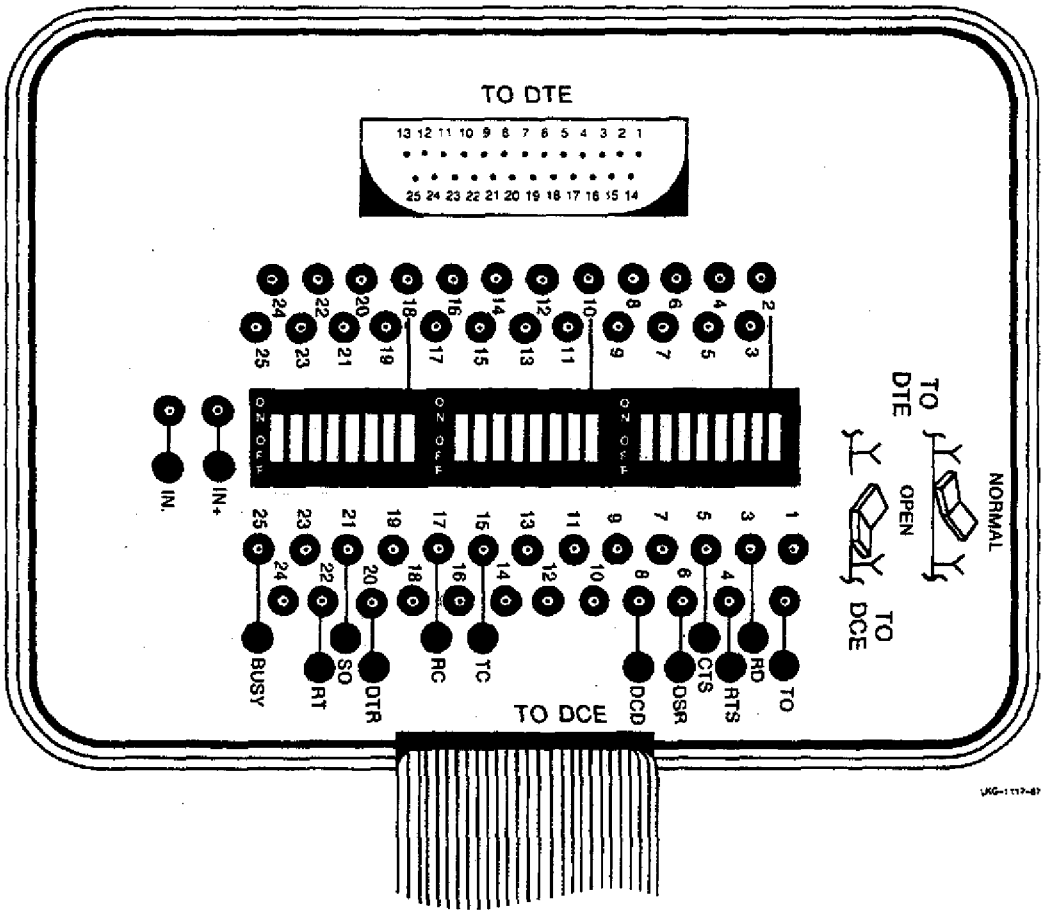
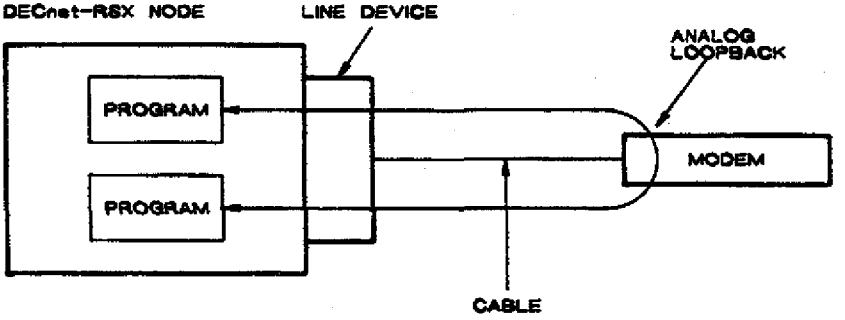


Figure 4–16: External Loopback Test Using a Modem



4.3.1 Line Level External Loopback Tests Using a Modem

This test checks the logic of the device transmitter and receiver, the line driver, the modem cable, and part of the modem. Figure 4-16 illustrates a line level external loopback test using a modem.

To perform this test, first set the line to SERVICE state and the controller mode to CONTROLLER NORMAL. Then issue the LOOP LINE command. For example:

```
NCP> SET LINE SDP-0 CONTROLLER NORMAL STATE SERVICE RET
NCP> LOOP LINE SDP-0 COUNT 10 RET
```

If this test is successful and you still have errors, contact your network authority. If the test fails, perform the tests described in the following sections to isolate the problem.

4.4 Tracing

The tracing facility can be used only on RSX–11 PSI systems. This facility records activity between the local DTE and DCE. Records are created for transmitted and received packets and frames at X.25 levels 2 and 3 respectively, and are placed in one or more specified files. X.25 level 2 defines link access procedures, and X.25 level 3 defines packet level procedures.

You can enable and disable tracing, specify where tracing takes place, and designate the output file by means of the NCP commands SET TRACE and CLEAR TRACE. For example,

```
NCP> SET TRACE LINE KMX-1-0 FILE BARRY [RET]
```

causes trace information from line KMX-1-0 to be placed in file BARRY.SYS. For a complete description of the NCP commands to perform tracing, refer to the *DECnet-RSX Guide to Network Management Utilities* .

The X.25 trace interpreter task (TRI) processes a file, or set of files, of trace records that have been created by the tracing facility. Through the use of command lines, TRI allows you to process these records and select, format, and print only the trace records you require for your particular trace analysis.

For a description of the TRI utility program, refer to the *DECnet-RSX Guide to Network Management Utilities* .

4.5 Dumping KMX/KMY/KMV (KMS-11) Microcode

This section describes how to dump the KMX/KMY/KMV (KMS-11) microcode to a file and how to analyze the dump file. The KMS-11 is a controller that is combined with X.25 level 2 microcode to provide a synchronous interface (KMX, KMY, or KMV) that is supported by RSX–11 PSI.

Use the NCP KMX-DUMP command to dump the microcode of the specified KMX, KMY, or KMV device to the file indicated. By default, the output file takes the following format, where *file-name* is the file that you specify.

file-name.DMP

For example, the following command dumps the microcode of the file BARRY.DMP in your current default directory:

```
NCP> KMX-DUMP LINE KMX-0-0 FILENAME BARRY [RET]
```

Use this command only if you believe there is an error in the microcode of the KMX, KMX, or KMV. The KMX/KMY/KMV (KMS-11) microcode dump analyzer (KDA) processes a dump file created by the NCP KMX-DUMP command. Through the use of command lines, KDA allows you to process this file and select, format, and print only the parts of that dump you require.

For a description of the KDA utility program, refer to the *DECnet-RSX Guide to Network Management Utilities* .

5

DECnet-RSX Host Services

DECnet-RSX can act as host node for a remote system. A host node in a DECnet network can perform the following operations:

- Down-line load an operating system image of a network node. See Section 5.1.
- Up-line dump the memory from a network node. See Section 5.2.
- Down-line load task images, overlay task memory segments, and provide checkpointing capabilities for tasks running on remote RSX-11S nodes. See Section 5.3.
- Connect to a remote console so that the local terminal emulates the console terminal of the remote node. See Section 5.4.

A host node can be a primary host node or a backup host node. Backup host nodes perform the host function if the primary host is unavailable.

Each section in this chapter has four parts:

- **Operation** - Describes the operation of the host service.
- **Requirements** - Describes requirements that must be met at both the host and remote nodes in order for the host service operation to be successful.
- **Commands** - Describes NCP commands used to set up or perform the service.
- **Parameters** - Describes parameters that are used for the host service.

5.1 Down-Line Loading

DECnet-RSX allows you to down-line load an operating system image. Down-line loading is the transferring of a copy of the system image file of a remote node's operating system from a network host node to the remote node. For example, DECnet-RSX permits you to load an RSX-11S operating system image file from your DECnet-RSX host node down-line to a remote node. Down-line loading can be initiated by a DECnet-RSX operator or by the remote node. Both procedures are discussed in this section.

To understand down-line loading, it helps to distinguish the nodes involved in the loading sequence. In the following node descriptions, the command node and the executor node can be the same or different nodes, but cannot be the target node.

- **Command Node.** An operator-initiated down-line load request originates at the command node. You use either the NCP LOAD or TRIGGER command to initiate this request.
- **Executor Node.** The executor node actually performs a down-line load or trigger operation. It must be adjacent to the target node because the down-line load is performed using circuit level access rather than logical links.
- **Target Node.** The target node is the node that receives the bootstrap loaders and the system image file. A remotely initiated down-line load request originates at the target node.

5.1.1 Down-Line System Load Operation

Down-line loading is initiated in one of two ways:

- **Target-initiated.** The target node initiates the operation by triggering its bootstrap ROM and sending a program load request to the executor node.
- **Operator-initiated.** An operator on the executor node initiates the operation with the NCP LOAD or TRIGGER command.

DECnet uses the Maintenance Operation Protocol (MOP) to perform down-line load operations. See the *Maintenance Operation Protocol Functional* Specification for a complete description of MOP. MOP is the protocol used by the Network Management layer of DNA to perform maintenance functions such as circuit testing, triggering, up-line dumping, and down-line loading.

5.1.1.1 Target-Initiated Down-Line Loads

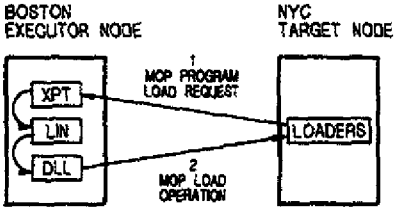
If the load is target-initiated and the circuit is a non-Ethernet circuit, the Routing layer (XPT) process detects MOP messages coming over the circuit and calls the line watcher (LIN). On Ethernet circuits, the Ethernet Protocol Manager (EPM) process detects the MOP messages and calls LIN. In both cases, LIN then changes the owner of the circuit from the Routing layer (XPT) process to the direct line access controller (DLX) process and calls the down-line system loader (DLL). DLL searches the remote node database for a node entry with a service circuit that matches the circuit over which the message was received, and, if it is an Ethernet circuit, an Ethernet address that matches the Ethernet source address from which the message was received. DLL then performs the load operation. Figure 5–1 illustrates this loading process. If the target on an Ethernet circuit does not have a specific host node from which to request a program load (for example, if the target's host node crashes) or if the load is initiated by means of the BOOT button on the target, the target proceeds as follows:

1. The target node sends a program load request message to the Ethernet load assistance multicast address AB-00-00-01-00-00: described in Section 2.1.6. This message is a request for any node on that Ethernet to perform the load.

2. The nodes on the Ethernet whose Ethernet circuits are enabled for service operations, check their own node databases for a remote node entry. A node whose Ethernet hardware address matches the hardware address of the target node, determines that the target can be down-line loaded. If down-line loading is possible, the secondary loader is sent to the target if the target requests the secondary loader, or if a message volunteering to do the load is received, that is, if the target is requesting the tertiary loader or operating system.

3. The target chooses the node responding first to continue the loading sequence. It does not send a message to any other node. The loading sequence continues normally from there.

Figure 5–1: Target-Initiated Down-Line Load



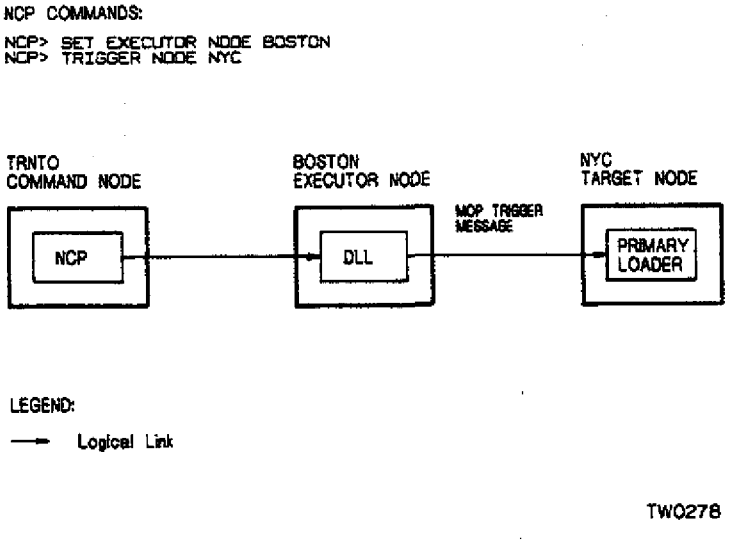
5.1.1.2 Operator-Initiated Down-Line Load

An operator-initiated load using NCP directly requests DLL to perform the load operation. The target node's primary bootstrap may or may not have to be triggered depending on the state of the target. The target node is triggered primarily to put it into a known state and to force it to supply program request information. Figure 5–2 illustrates the loading process.

Use the NCP LOAD or TRIGGER command to perform an operator-initiated down-line load. The TRIGGER command allows you to directly trigger the remote node's bootstrap ROM, if the device supports it, which causes the target node to load itself according to how the primary loader is programmed to operate. The programs to be loaded may come from a disk file local to the target node, another adjacent node, or the executor node.

Note that the TRIGGER command may or may not initiate a down-line load. One of the functions of this command is to simulate the operation that occurs when the BOOT button on the target node is pushed. A bootstrap operation from a local disk may result.

Figure 5–2: Operator-Initiated Down-Line Load



When you use the LOAD command, the executor node proceeds with the load operation according to the options specified in the initial load request. Any required information that has been defaulted is obtained from the volatile database. With this information, the executor is thereby able to control the load sequence.

Section 5.1.3 describes the TRIGGER and LOAD commands, their parameters, and examples of their use.

5.1.1.3 Load Sequence

The load sequence is the same, regardless of whether the request was initiated by the operator or the target node. The first program to run at the target node is the primary loader. Typically, this program is either executed directly from the target node's bootstrap ROM, or it is in the microcode of the load device: DMC, DMR, DMP, DMV, UNA, or QNA. Once the target node's primary loader is triggered, the target node sends a REQUEST PROGRAM LOAD message to the executor node requesting a program load. Usually, the primary loader requests a secondary loader program, which, in turn, may request a tertiary loader. The final program to be loaded is the operating system. In this sequence, each program requests the next one until the operating system is loaded. After the load sequence, the target receives a message with the name of the host and places the name in its volatile database. The target can then use the host node for down-line task-loading applications.

The secondary loader is small, 400-600 bytes, and is always sent as a single MEMORY LOAD WITH TRANSFER message. The secondary loader is always loaded into the lower 32K words of memory and operates without memory management. The secondary loader requests the tertiary loader which is sent in multiple MEMORY LOAD messages followed by a MEMORY LOAD WITH TRANSFER ADDRESS message. The tertiary loader is much larger with 1K-3K bytes and uses memory management to relocate itself to the top of memory before requesting the operating system. Therefore, any operating system being loaded must not use the top 3K bytes of memory or the tertiary loader overwrites itself and the load will not be successful. The operating system is sent in multiple MEMORY LOAD messages followed by a PARAMETER LOAD WITH TRANSFER ADDRESS message. This supplies the start address of the image just loaded and contains extra values for the node identification and the host identification. See Section 5.1.4.2.

5.1.2 Down-Line Load Requirements

Before attempting a down-line load operation, you must ensure that the nodes, lines, and circuits involved in the down-line load meet the following requirements:

- The target node must be connected directly to the executor node. The executor node uses circuit level access to load the target system; therefore the target node must be adjacent to the executor node because no routing information is used.
- The proper bootstrap must be present at the target node. See Section 5.1.2.1. This bootstrap must be capable of recognizing TRIGGER messages if the node is to be operator-initiated down-line loaded, or it must be able to send program requests if the load is target-initiated.

- The physical hardware devices must be set up correctly to support the load. See Section 5.1.2.2.
- For target-initiated loads only, the host node circuit involved in the load operation must be enabled to perform service functions. See Section 5.1.2.3.
- The executor node must have access to the load files. The location of the files can be either specified in the load request or defaulted to in the remote node database. See Section 5.1.4.3.

5.1.2.1 Down-Line Load Bootstraps

The down-line load sequence is started by a primary loader. This primary loader is contained in the target node's bootstrap ROM or in the microcode of the target node's load device. In either case you must ensure that you have the proper ROM support in order for the down-line load to be initiated. The devices used for down-line loading and the ROMs that support these devices are listed in Table 5–1. This table also lists the type of bootstrap supported by each device:

- **Power-on boot.** This device can be used when the bootstrap will be invoked by a power-on condition.
- **Remote trigger detect boot.** This device can be used for operator-initiated down-line loads because it responds to the TRIGGER message.
- **Console boot.** A power-on boot cannot be used; however the target's console can be used to start the down-line load sequence. If this is the only type of boot supported, the target system cannot operate entirely unattended.

Table 5–1: Down-Line Load Bootstrap Support

Device	Remote Load Detect	Power-On Boot	Console Boot
DMP11	1	1	N/A
DMV11	1	1	N/A
DEUNA/DELUA	1	1	2
DMC11	3	3	3
DMR11	3	3	3
DL11-E	N/A	3	3

(continued on next page)

Table 5–1 (Cont.): Down-Line Load Bootstrap Support

Device	Remote Load Detect	Power-On Boot	Console Boot
DU11	N/A	3	3
DUP11	N/A	3	3
DLV11-E/F	N/A	4	4
DUV11	N/A	4	4
DEQNA/DELQA	N/A	5	5

1 Device configuration (switch settings) required
2 M9312 required
3 M9301 or M9312 required
4 BDV11 required
5 KDF11-B2
N/A Not Available

5.1.2.2 Setting Up the Down-Line Load Devices

In order for the down-line load sequence to function correctly, the devices involved in the down-line load operation must be configured properly. See the operations manual for the device being used.

It is imperative that the control status register (CSR) and vector of the target's down-line load device be set up properly. The primary bootstraps either expect the device in a fixed location or use a floating CSR algorithm to calculate the CSR of the device. This floating CSR calculation can be performed using the FLOAT.CMD procedure found in UIC [200,200] on the distribution kit.

Whenever possible you should verify the operation of the target's and the host's down-line load device before trying the down-line load.

5.1.2.3 Setting Up the Host Circuit

The host circuit must be in the ON or SERVICE state; an Ethernet circuit must be in the ON state. For example, the following command readies circuit DMC-0 for down-line loading an adjacent node:

```
NCP> SET CIRCUIT DMC-0 SERVICE ENABLED STATE ON RET
```

5.1.3 Down-Line Load Commands

The most convenient method of down-line loading is to set up default information in the volatile database. You can use the NCP or CFE command SET or DEFINE NODE to establish default information for the target node in the volatile or permanent database. See Section 5.1.4 for a description of the parameters that may be set up. These default parameters are also used for target-initiated down-line loads, although the MOP program load message can override some of the defaults. This default method is discussed later in this chapter. Alternatively, you can override the default by specifying several parameters for the NCP TRIGGER or LOAD command. This section describes each command and illustrates its use.

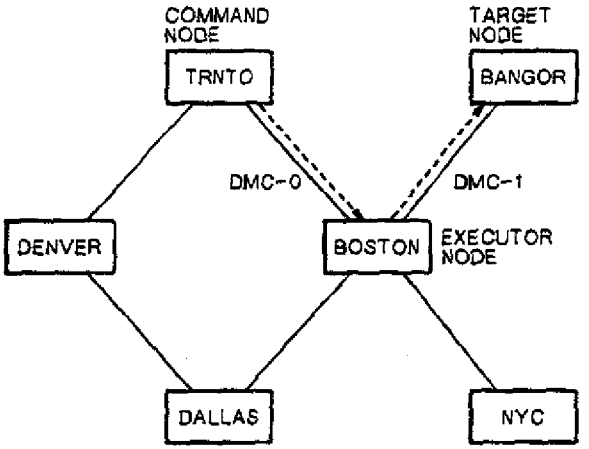
5.1.3.1 TRIGGER NODE and TRIGGER VIA Commands

The TRIGGER command triggers the bootstrap mechanism of a target node, which causes the node to request a down-line load. Because the system that is being booted is not necessarily a fully functional network node, the operation must be performed over a specific circuit. To bring up the system at the target node, use either the TRIGGER NODE or TRIGGER VIA command. If you use the TRIGGER NODE command and do not specify a loading circuit, the executor node obtains the circuit identification associated with the target node from its volatile database. If you use the TRIGGER VIA command, which indicates the loading circuit but not the node identification, the executor node uses the default target node identification in its remote node database. To establish the default circuit to be used for loading, use the SET or DEFINE NODE command with the appropriate SERVICE CIRCUIT parameter. See Section 5.1.4.1. To establish the default node identification for a target node, use the SET or DEFINE NODE command with the appropriate NODE and NAME parameters. See Section 5.1.4.2.

The following command triggers node BANGOR in the example shown in Figure 5–3.

```
NCP> TRIGGER NODE BANGOR VIA DMC-0 RET
```

Figure 5–3: Operator-Initiated Down-Line Load over a DDCMP Circuit (NCP TRIGGER)



NCP Commands:

```
NCP> SET EXECUTOR NODE BOSTON
NCP> SET LINE DMC-1 ALL
NCP> SET CIRCUIT DMC-1 STATE ON
NCP> TRIGGER NODE BANGOR SERVICE PASSWORD FEFEFEFE
```

TWO250

If you choose to override the default parameters that were set up in the remote node database using the NCP or CFE command SET or DEFINE NODE, you can change the following aspects of the TRIGGER sequence:

- The physical address of the target node. See Section 5.1.4.2.

  PHYSICAL ADDRESS *ethernet-address*

- The service password needed to trigger the target's load device. See Section 5.1.4.2.

  SERVICE PASSWORD *password*

- (TRIGGER NODE only) The service circuit used for the down-line load. See Section 5.1.4.1.

  VIA *circuit-id*

When issuing the TRIGGER or TRIGGER VIA command, you can specify any or all of the parameters listed previously. Any parameters not specified in the command default to whatever information is specified in the node database. Use the SET or DEFINE NODE command to establish default node database information.

NOTE

Once the target node is triggered, it will then load itself in whatever manner its primary loader is programmed to operate. The target node could request a down-line load from either the executor that just triggered it or another adjacent node, or the target node could load itself from its own mass storage device.

5.1.3.2 LOAD NODE and LOAD VIA Commands

Use the LOAD NODE and LOAD VIA commands to load software down-line to a target node. The LOAD NODE and LOAD VIA commands will work only if the target node can be triggered by the executor or if the target has been triggered locally. See Section 5.1.3.1.

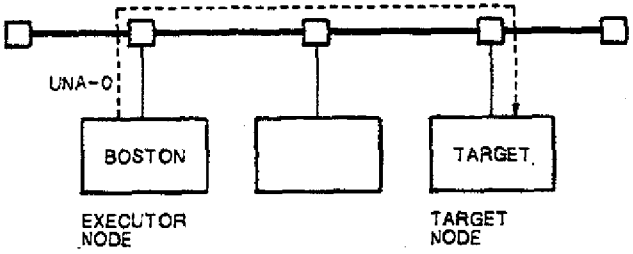
The LOAD NODE command requires the identification of the service circuit over which to perform the load operation. If you do not explicitly specify a service circuit in this command, the executor node uses the SERVICE CIRCUIT from the remote node database entry for the target node. You must use the SET or DEFINE NODE command to include the SERVICE CIRCUIT entry in the remote node database. Alternatively, you can explicitly include the circuit in the command (LOAD NODE BANGOR VIA DMC-0).

You can also use the LOAD VIA command to specify the circuit over which to perform a down-line load. For example, to load using circuit DMC-0 connected to the executor node, issue the command

```
NCP> LOAD VIA DNC-0 RET
```

In both types of LOAD commands, the executor node obtains the rest of the necessary information from its remote node database. Figure 5–4 shows a LOAD request over an Ethernet circuit.

Figure 5–4: Operator-Initiated Down-Line Load over an Ethernet Circuit (NCP LOAD)



NCP Commands:

```
NCP> SET NODE TARGET SERVICE CIRCUIT UNA-0
NCP> SETT CIRCUIT UNA-0 STATE ON
NCP> LOAD NODE TARGET PHYSICAL ADDRESS AA-00-04-00-C8-04
```

TWO249

If you choose to override the default parameters for the LOAD commands, you can change the following aspects of the load sequence:

- (LOAD NODE only) The service circuit over which the load is to take place. See Section 5.1.4.1.

  VIA *circuit-id*

- The host node that the target node is to use once the target is loaded. See Section 5.1.4.2.

  HOST *node-id*

- The identification of the load files. See Section 5.1.4.3.

FROM *file-id*

SECONDARY LOADER *file-id*

TERTIARY LOADER *file-id*

DIAGNOSTICS FILE *file-id*

- The DECnet Phase of the target node. See Section 5.1.4.2.

  SERVICE NODE VERSION *phase*

- The identification of the target node's line device type that is to handle service operations. See Section 5.1.4.2.

  SERVICE DEVICE *device-type*

- The identification of the service password for triggering the target node's bootstrap mechanism. See Section 5.1.4.2.

  SERVICE PASSWORD *hex-password*

- For Ethernet targets, the physical address of the target. See Section 5.1.4.2.

  PHYSICAL ADDRESS *ethernet-address*

When issuing the LOAD NODE and LOAD VIA commands, you can specify any or all of these parameters. Any parameter not specified in the command defaults to whatever information is specified in the remote node database. Use the SET or DEFINE NODE command to establish default information for the target node's parameters in the remote node database.

5.1.4 Down-Line Load Parameters

You can define default parameters for down-line loading in either the permanent or volatile database. Each target system to be loaded has a separate set of defaults. The defaults for a specific target apply whether the load is operator-initiated or target-initiated. Use the DEFINE or SET NODE command to set up the database. To remove node parameters from the database, use the PURGE or CLEAR NODE command. Note that some parameters have a different keyword associated with them. The parameter keyword depends on whether the parameter is being specified in the SET or DEFINE NODE command or in one of the LOAD or TRIGGER commands. The SET NODE's LOAD FILE parameter is specified as the FROM parameter in the LOAD command.

The down-line load parameters that you can define for each target in the database include:

- A service circuit parameter for the executor node

  SERVICE CIRCUIT *circuit-id*

- Parameters that pertain to the target node

  NODE *node-id*

  NAME *node-name*

  SERVICE NODE VERSION *phase*

  SERVICE DEVICE *device-type*

  SERVICE PASSWORD *password*

  HARDWARE ADDRESS *ethernet-address* (Ethernet nodes only)

  HOST *node-id*

  Note that the NODE, NAME, and HOST parameters used for the load are included in the MOP message that the executor sends to the target when the load completes. See Section 5.1.1.

- Parameters that specify load files

  LOAD FILE *file*

  SECONDARY LOADER *file*

  TERTIARY LOADER *file*

  DIAGNOSTIC FILE *file* (Ethernet Communications Server only)

All of these parameters are described in the following sections.

NOTE

In order for a down-line load request to be successful you must specify at least the SERVICE CIRCUIT, NODE, NAME, and LOAD FILE parameters either in the LOAD command or by previously defined defaults. For Ethernet nodes,

the HARDWARE or PHYSICAL ADDRESS must also be specified.

5.1.4.1 The Service Circuit Parameter

The following parameter specifies a circuit that links the executor node to the adjacent target node:

- SERVICE CIRCUIT is the parameter that specifies the circuit over which the load operation is to occur. In the TRIGGER NODE and LOAD NODE commands, this parameter is specified with the VIA parameter. In the TRIGGER VIA and LOAD VIA commands, this parameter is always explicitly specified in the command and causes a search of the node database for a node with the specified service circuit and physical address for Ethernet circuits.

Format:

TRIGGER VIA *circuit-id* PHYSICAL ADDRESS *ethernet-address*

[SERVICE] PASSWORD *password*

Example:

```
NCP> TRIGGER VIA DNC-0 [RET]
```

This command triggers the bootstrap mechanism on the node connected to circuit DMC-0.

5.1.4.2 The Target Node Parameters

The following parameters relate to the target node:

- NODE identifies the target node by name or address. You can set up a separate down-line load database entry in the permanent database for each target node to be down-line loaded.
- NAME associates a node name with a target node address specified in the previous parameter.
- SERVICE NODE VERSION describes the target node as either Phase III or Phase IV. If you do not define this parameter either in the database or in a command, it defaults to Phase IV. When using the LOAD command, you can override any default information by using the SERVICE NODE VERSION parameter.

- SERVICE DEVICE specifies the controller on the target node end of a service circuit. The service device handles down-line loading in a variety of ways, depending on the device used. The SERVICE DEVICE parameter identifies the default secondary and tertiary loaders for the load operation. See the description of secondary and tertiary load files in Section 5.1.4.3. This parameter is required for any down-line load if the secondary and tertiary load files are not specified in the remote node database. SERVICE DEVICE is also required if the program load requests from the target node do not specify the secondary and tertiary load file names. For example, the following command identifies the service device as a DMC device controller.

  ```
  NCP> SET NODE BANGOR SERVICE DEVICE DMC [RET]
  ```

  When using the LOAD command you can override any default information by using the SERVICE DEVICE parameter.

- SERVICE PASSWORD specifies a default service password, required for some devices to trigger the primary bootstrap mechanism on the target node. Depending on the service device being used, the password must be a hexadecimal number in one of two ranges. For DMC, DMR, DMP, or DMV devices, the range is 0 to FFFFFFFF. The hardware devices contain switches for setting up two hexadecimal digits as the password which are then repeated four times to make up the actual password; for example , 3 on the device would be entered as: 03030303 as specified by CFE/NCP. For UNA devices, the range is 0 to FFFFFFFFFFFFFFFF. For example, the following command sets the service password for a node with a load device in the first category:

  ```
  NCP> SET NODE BANGOR SERVICE PASSWORD FEFEFEFE [RET]
  ```

  When using the LOAD command, you can override any default information by using the SERVICE PASSWORD parameter.

- HARDWARE ADDRESS specifies the Ethernet hardware address originally assigned to the controller (the DEUNA, DEQNA, DELUA, or DELQA) for the target node. This address identifies an Ethernet node before it has started running DECnet and has set a logical address for itself, called the physical address. See Section 2.1.6 for a description of Ethernet addressing. You can define the hardware address in either the permanent or the volatile database, but not in a LOAD or TRIGGER command. The corresponding LOAD or TRIGGER command parameter is PHYSICAL ADDRESS.

  If you do not specify a PHYSICAL ADDRESS parameter in a LOAD or TRIGGER command, the HARDWARE ADDRESS must be defined in the node database. In this case, DECnet-RSX attempts to use that hardware address to communicate with the target node. If the attempt fails,

DECnet-RSX tries again using the physical address it has created from the target's node address.

- HOST specifies the node that will load task files down line to the target node. See Section 5.3 for information about down-line task loading. The host can be the executor node (the default) or any node other than the target node itself. At the end of the load sequence, the target receives a message with the name of the host and places that name in its volatile database. See Section 5.1.1.3. The target can then use the host node for down-line task-loading applications. In addition, RSX tasks running on the target can issue connect requests to $HOST. For example, the following command sets the host to node NYC once node BANGOR comes up as a network node if BANGOR has the necessary DECnet software:

  ```
  NCP> SET NODE BANGOR HOST NYC [RET]
  ```

  When using the LOAD command, you can override any default information by using the HOST parameter.

5.1.4.3 Down-Line Load Files

The load files are those files that are to be down-line loaded to the target node during the load sequence. These files may consist of a secondary loader, a tertiary loader, the operating system image, and a diagnostic file. All load files default to *dv*:[*netuic*]*file-name*.SYS where *dv*: is the device the network tasks are loaded from, and *netuic* is the network UIC. The following parameters specify various load files:

- LOAD FILE specifies the file that contains the system image for loading down line to the target. When using the LOAD command, you can override any default information by using the FROM parameter.
- SECONDARY LOADER and TERTIARY LOADER specify files containing programs that are part of the load sequence. If the node database contains secondary and tertiary file name entries, they are the LOAD command defaults for the SECONDARY and TERTIARY parameters. However, if the volatile database does not contain entries for these parameters and if you omit them from a LOAD command, the executor selects a secondary and a tertiary file according to the type of service device specified for the target. The executor chooses secondary and tertiary loader files from the list in Table 5–2.

  If, however, you include the secondary and tertiary file names as entries in the remote node database for the target node, they can override the default loader files just described. By using the SET or DEFINE NODE command, you can select your own special load files for a particular target node. If you do not specify default loader files or a default service device,

you can change the service device type at the target node without having to change the target node's database entry at the executor node.

NOTE

If the target node being loaded is a Phase III node, the secondary and tertiary loader files must be obtained from a Phase III kit.

- DIAGNOSTIC FILE (Ethernet Communications Server only) specifies a file containing diagnostics that the target server system can request. The use of this diagnostics image is documented in the manuals provided with the server.

Table 5–2: Default Loader Files by Target Device Type

Device Type	Secondary Loader	Tertiary Loader
DLV	SECDLV.SYS	TERDLV.SYS
DL11	SECDL.SYS	TERDL.SYS
DMC11	SECDMC.SYS	TERDMC.SYS
DMP	SECDMP.SYS	TERDMP.SYS
DMV	SECDMV.SYS	TERDMV.SYS
DPV	SECDPV.SYS	TERDPV.SYS
DU11	SECDU.SYS	TERDU.SYS
DUP11	SECDUP.SYS	TERDUP.SYS
DUV11	SECDUV.SYS	TERDUV.SYS
QNA	SECQNA.SYS	TERQNA.SYS
UNA	SECUNA.SYS	TERUNA.SYS
LUA	SECLUA.SYS	TERLUA.SYS
LQA	N/A	TERLQA.SYS

5.2 Up-Line Dumping of Memory

As a DECnet-RSX network manager, you can include certain node parameters in the remote node database that will allow an adjacent node to dump its memory into a file on your DECnet-RSX system. This procedure is referred to as up-line dumping. It is a valuable tool for crash analysis; that is, system programmers can analyze the dump file and determine why the remote system failed. When an adjacent node that has this feature included detects an impending system failure, that system will request an up-line dump. For example, an RSX–11S system may request an up-line memory dump to a DECnet-RSX system when it senses a system failure.

For up-line dump operations, the local DECnet-RSX node is referred to as the executor node and the adjacent unattended node as the satellite node.

5.2.1 Up-Line Dump Operation

This section describes the procedures for an up-line dump initiated by a satellite node. Up-line dumping, unlike down-line loading, is always initiated by the satellite node; there are no NCP commands to operator-initiate an up-line dump. DECnet uses the Maintenance Operation Protocol (MOP) to perform an up-line dump operation. MOP is a protocol used for circuit testing, triggering, down-line loading, and up-line dumping. Refer to the *Maintenance Operation Protocol Functional* Specification for a complete discussion.

5.2.1.1 Up-Line Dump Sequence

There are four steps involved in the up-line dump process. The actual dump takes place by repeating step 3.

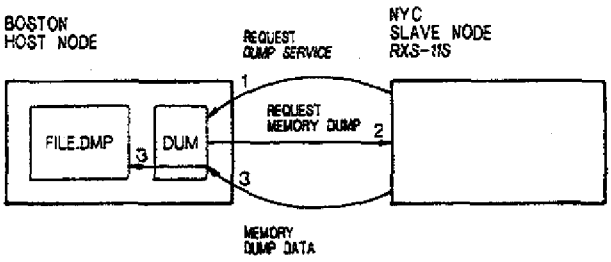
1. When a satellite node senses a system failure, it sends a REQUEST DUMP SERVICE message to its host node or, on Ethernet, to a dump assistance multicast address if the Ethernet host is not available. See Figure 5–5. This message might contain information about the satellite's memory size and the up-line dump device type at the satellite.
2. If the message from the satellite includes a memory size value, the host node uses it. NETPAN on RSX–11S always provides this. Otherwise, the host node checks the satellite node's entry in its remote node database for the DUMP COUNT value. The host also retrieves the memory address from which to start dumping and the file where the memory image will be stored. The default is 0. The host node, which can now be considered the executor, sends a MOP REQUEST MEMORY DUMP message to the satellite with the starting address and buffer size values.

3. Using the values it receives from the executor, the satellite returns the requested block of memory in a MOP MEMORY DUMP DATA message. The executor receives the block of dump data, places it in the dump file, increments the memory address by the number of locations sent, and sends another REQUEST MEMORY DUMP message to the satellite. This sequence is repeated until the amount of memory dumped matches the memory size specified in the dump request or retrieved from the node database.

4. Once the up-line dump is completed, the executor node automatically attempts to down-line load the satellite system. It initiates the down-line load by sending a TRIGGER message to the satellite. See Section 5.1.1.2.

Figure 5–5 illustrates the up-line dump procedure.

Figure 5–5: Up-Line Dumping a Remote Node



TWO279

If the satellite node is on an Ethernet circuit, the satellite will attempt to perform an up-line dump to the node which originally loaded it down-line. If that node is not available, the satellite node proceeds as follows:

- The satellite node sends a memory dump request to the Ethernet dump assistance multicast address AB-00-00-01-00-00. This message is a request for any node on the Ethernet to receive an up-line memory dump.
- The nodes on the Ethernet check their own databases to determine if they can accept an up-line dump. For Ethernet circuits, the nodes look for a circuit and physical address match. If so, they respond to the satellite node. The satellite chooses the node responding first to continue the dumping sequence. The satellite does not send a message to any other node. The dumping sequence continues normally from there. Note, however, that you may have to look for event 0.3 in the event logs for all nodes on the Ethernet to determine which node received the dump. An alternative way to use the event logging facility is to use remote event logging.

5.2.2 Up-Line Dump Requirements

Before attempting an up-line dump operation, you must ensure that the nodes, lines, and circuits involved in the up-line dump operation meet the following requirements:

- The satellite node must be directly connected to the executor node by a physical line. The executor node uses circuit level access to dump the satellite node; therefore, the satellite node must be adjacent to the executor because no routing information is used.
- The satellite node must be capable of requesting the up-line dump when it detects a system failure. If the dumping program does not exist on the satellite, up-line dumping cannot occur. See Section 5.2.2.1.
- The circuit involved in the dump operation must be enabled to perform service functions. See Section 5.2.2.2.
- If the satellite does not supply the memory size value, the executor must have this value in its remote node database. See Section 5.2.3.
- The executor must have a dump file specified in the remote node database and must be able to create this file. See Section 5.2.3.

5.2.2.1 Including Up-Line Dump Support in the Satellite Node

Including up-line dump support in the satellite node requires procedures specific to the system being dumped up-line. For RSX-11S systems, the procedure for including up-line dump support is documented in the *DECnet-RSX Network Generation and Installation Guide*. For Ethernet Communications Server products, see the individual product documentation.

5.2.2.2 Setting Up the Host Circuit

The host circuit must be in the ON and SERVICE ENABLED state. For example, the following command readies circuit DMC-0 for up-line dumping:

```
NCP> SET CIRCUIT DMC-0 SERVICE ENABLED STATE ON  RET
```

5.2.3 Up-Line Dump Parameters

You can define default parameters for up-line dumping in either the permanent or volatile database. Each satellite system to be dumped has a separate set of defaults. Use the SET or DEFINE NODE command to set up the default information. To remove node parameters from the node database, use the CLEAR or PURGE NODE command. In order for an up-line dump request to be honored, you must specify the DUMP FILE parameter.

- DUMP FILE specifies the name of the file that will hold the up-line dump image. The executor must be able to create this file when the up-line dump request is received.
- DUMP COUNT specifies the default amount of memory that the executor requests from the satellite node. If this value is not specified, it must be supplied by the satellite node. You do not have to set this value to be equal to the memory size of the satellite; partial memory dumps are allowed. If the satellite node supplies this parameter in the dump request, the value received overrides any default you set with the SET or DEFINE NODE command.
- DUMP ADDRESS specifies the starting address for the up-line dump. If this value is not specified, the default is to start at address 0.

5.3 Down-Line Loading RSX-11S Tasks

Down-line task loading extends nonresident initial task load, checkpointing, and overlay support to a DECnet-11S node. You can load an RSX-11S task down line by using the Satellite Task Loader (SLD) on the DECnet-11S node and the Host Task Loader (HLD) on the DECnet-RSX node.

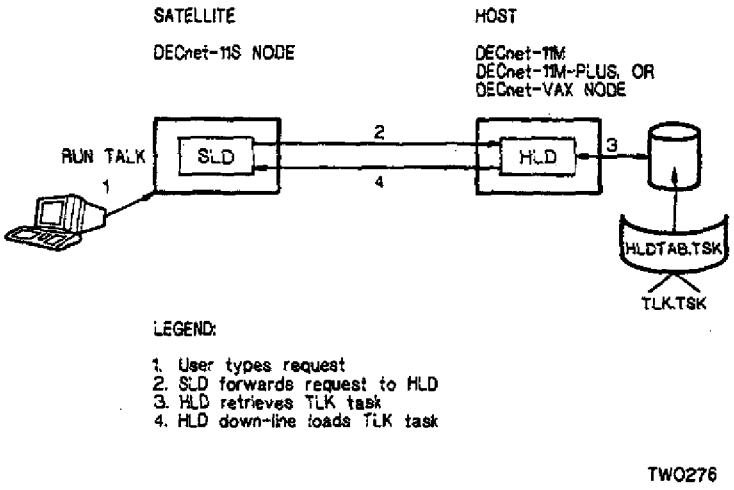
For down-line task loading operations, the host node is referred to as the host and the target node as the satellite.

5.3,1 Down-Line Task Loading Operation

SLD uses the intertask communication facilities of DECnet–11S to communicate with HLD. Figure 5–6 illustrates one instance of this relationship using the example of the TLK utility.

A user on the RSX–11S node types RUN TLK. The SLD task sends a task image request to the host HLD task. The HLD task looks for the TLK entry in the HLDTAB.TSK database. Using the information found in the database, HLD sends the task image data to the SLD task on the RSX–11S node.

Figure 5–6: Down-Line Task Loading an RSX–11S Task



5.3.2 Down-Line Task Loading Requirements

Before attempting a down-line task load, you must ensure that the following requirements are met:

- The host node must be available. Unlike down-line loading and up-line dumping, it is **not** a requirement that the host node be adjacent to the satellite node. Down-line task loading is performed using normal DECnet logical links and, therefore, routing information is available.
- The satellite node must have the SLD task installed properly. See Section 5.3.2.1.
- Any tasks to be down-line loaded must be installed in the satellite system image. See Section 5.3.2.1.
- The host node must have an entry in its mapping table for the requested task. See Section 5.3.2.2.

5.3.2.1 Setting Up the Satellite System

You build the SLD task during the RSX–11S NETGEN procedure. Refer to the *DECnet-RSX Network Generation and Installation Guide*. To allow down-line task loading, use the appropriate VMR commands to install and fix SLD into the RSX–11S system. For example:

```
> VMR [RET]
Enter filename: RSX11S [RET]
VMR> INS SLD [RET]
VMR> FIX LDR... [RET]
```

This sequence of commands establishes SLD as the loading task (LDR...) for the executive.

Any tasks to be down-line loaded to or checkpointed from the RSX–11S system must be installed, but not fixed, using VMR. For example:

```
> VMR [RET]
Enter filename: RSX11S [RET]
VMR> INS TLK [RET]
```

In this example, entering RUN TLK on a terminal of the RSX–11S remote system initiates the down-line task load of the file TLK.TSK.

If the RSX–11S system will not be loaded down-line, you must specify the node to which SLD will connect, using the VNP command SET EXECUTOR HOST. See the *DECnet-RSX Guide to Network Management Utilities* . For example, you could use the following command, where 11 is the number of the node BOSTON on which HLD resides:

```
> VNP RSX11S [RET]
VNP> SET EXECUTOR HOST 11 [RET]
```

5.3.2.2 Setting Up the Host System - Using the Host Task Loader (HLD)

The Host Task Loader utility (HLD) is the down-line task loader service task that runs on the host system. HLD communicates with the Satellite Task Loader utility (SLD) running on a remote RSX–11S system to enable tasks on the host system to be down-line loaded to the remote node. You build HLD during the host's network generation. For more information, see the *DECnet-RSX Network Generation and Installation Guide*.

HLD is driven by an external mapping table that maps requests from a satellite's SLD to a file residing on the host system. The mapping table is divided into two logical sections, each containing descriptors for a different type of task:

General purpose tasks are listed first in the table. They are not associated with any particular node and can be down-line loaded to any RSX–11S node in the network. Such tasks cannot be checkpointed. Guidelines for specifying and using general purpose tasks follow:

- You can specify the same task name more than once in the general purpose task list. This allows RSX–11S systems to share installed task names.
- Task mapping is automatically performed unless you specify otherwise.
- LUN fixing is automatically performed.
- You must install general purpose tasks in the same partition for all RSX–11S systems, although the partition addresses need not be identical.
- If you add new nodes to the network, existing general purpose tasks can be down-line loaded to those nodes without your having to change the mapping table. Nodes receiving only general purpose tasks do not have to be defined in the HLD mapping table.

Node-specific tasks are always preceded by a node name in the table and can be down-line loaded to that node only. You must specify LUN fixing if you wish this feature.

Creating and Modifying the HLD Mapping Table

To create or modify the external HLD mapping table image, execute the [*grp*,1]HLDDAT.CMD command file provided by NETGEN; *grp* is the group code specified during NETGEN. This command file leads you through the

steps required to specify your tasks. To get help at any time during the procedure, press ESC in response to a prompt.

The first prompt is for a command string. You can enter any of the following commands:

DEFINE	Creates a new node and/or task entry.
EXIT	Terminates the session, replacing the old database contents and optionally assembling and building the new database. EXIT is the same as CTRL/Z.
HELP	Displays a list of the HLD commands.
LIST	Displays all node and task entries.
PURGE	Deletes all old node and/or task entries.
QUIT	Terminates the session with no changes to the old database contents.

First you define any general purpose tasks that you wish to create or modify. Enter the task image file name in response to the prompt. You are then prompted to enter a task name if you wish it to be different from the task image file name. You can also override task mapping if you wish. When you have made all your general purpose task entries, press RET to begin node-specific task definitions.

If there are any nodes already specified, the system displays each node name in turn and asks if you want to define any new tasks for it. For each task that you define, you can choose whether or not you want LUN fixing performed. See the end of this section. Press RET when you are ready to define tasks for a different node. The system will either display the name of another node that has already been specified, or it will ask if you would like to specify a new node.

When you have made all your task specifications, enter EXIT and press RET. The system will then ask if you want to build the new HLD database now. If you choose to build the database later, you can either execute the HLDDAT.CMD command file again and issue EXIT or issue the following commands to create the task image file.

```
> SET /UIC=[100,24] RET
> MAC @HLDTABASN RET
> TKB @HLDTABBLD RET
```

LUN Fixing

LUN fixing is an SLD feature that reinitializes the LUNs after the task has been down-line loaded. This is equivalent to the user program issuing an LUN$ assignment at the start of the program. This feature allows a single task to be loaded into multiple RSX-11S systems that may have different unit control block addresses.

LUN fixing causes the task down-line load to be followed by a copy of the ASCII device assignments. SLD then dynamically assigns the loaded task's LUNs.

LUN fixing is automatic for general purpose tasks. You must specify it if you wish LUN fixing on a node-specific task.

HLD Error Handling

The HLD utility returns error messages resulting from file access and network communications operations and from inconsistencies in the mapping table and file to be loaded. The error messages for HLD are described in the *DECnet-RSX Guide to Network Management Utilities* .

5.4 Remote Console Facilities

The console carrier requester (CCR) allows you to set up a logical connection between your DECnet-RSX node and the console interface on certain remote nodes. This permits your terminal to act as the console for the remote system; for example, your terminal can act as the console for the Digital Ethernet Communications Server (DECSA) hardware and its resident software, such as the Router Server. The console carrier requester connects to a companion program called the console carrier server (CCS).

You can use the CCR to force a crash if the server node becomes unresponsive. To determine how to force a crash, see the applicable documentation for the given server product. CCR also permits debugging under special circumstances.

5.4.1 CCR Operation

The console carrier server of a remote node can be in one of three states when you wish to use the console:

- Loaded and unreserved
- Loaded and reserved
- Not loaded

If the console carrier server is loaded and unreserved, the CCR reserves it, and the following message is displayed on your terminal:

```
CCR -- Remote console reserved
```

If the console carrier server is loaded and reserved, an event message is displayed on your terminal. A complete list of CCR error messages is given in the *DECnet-RSX Guide to Network Management Utilities* .

If the console carrier server is not loaded, the CCR loads the server. It may be necessary for you to identify the target node by using the 12 hexadecimal digit Ethernet physical address if the target node is not known to the network. Once the server is loaded, the CCR attempts to reserve the console and proceeds as already described.

5.4.2 CCR Requirements

In order to use the CCR, you must be sure that the following conditions are met:

- The host node and the remote node are on the same Ethernet.
- The console carrier server image and its loader file are present in the network UIC on the host node. The file name of the console carrier server image is PLUTOCC.SYS and that of the loader is PLUTOWL.SYS.

5.4.3 CCR Commands

To invoke the CCR and connect to a remote console carrier server, use the following command:

CCR NODE *node-id* [SERVICE CIRCUIT *circuit-id*]
[SERVICE PASSWORD *password*]
[PHYSICAL ADDRESS *ethernet-address*]

where

NODE *node-id*	specifies the target node by name: 1 to 6 alphanumeric characters, including at least 1 alphabetic character or address in the range of 1 to 1023 for single area networks, or in the format *x.y*, where *x* is in the range 2 to 63, and *y* is a value from 1 to 1023.
SERVICE CIRCUIT *circuit-id*	identifies the circuit to the target node. The variable *circuit-id* has the format *dev-c*[*-u*][*.t*]. See Section 2.5.2.1.
SERVICE PASSWORD *password*	defines the password required to access the target node. The password is a hexadecimal number in the range of 0 to FFFFFFFFFFFFFFFF; maximum 16 hexadecimal digits.
PHYSICAL ADDRESS *ethernet-address*	identifies the Ethernet physical address that the node currently uses to identify itself. Required if the hardware address parameter has not been specified in the volatile database; see the SET NODE command in the *DECnet-RSX Guide to Network Management Utilities* .

The *node-id* is required. If the other parameters are not specified in the command line, they must be specified in the down-line load database. If they are specified in the command line, they override the parameters set in the down-line load database.

5.4.4 CCR Special Characters

Two special characters are used by the CCR:

CTRL/B	Operates as a BREAK command to get the attention of the console on-line debugging tool (ODT).
CTRL/D	Initiates an exit from console carrier mode and terminates the CCR.

5.4.5 Sample CCR Session

In the following sample session, you invoke the CCR to test node DALLAS. The system then displays the current program counter and a prompt (@). At this point, you can enter commands as if you were physically located at a console. The command is transmitted by the requester to the server, and the appropriate response is transmitted by the server to the requester. The requester then displays this information.

To examine a register, enter the dollar sign ($) followed by the register number and a slash (/). To examine a location, enter the address of the location and a slash. The system then displays the contents on the same line. Press LF to view the next register or location, or press RET to return to the @ prompt. When you wish to exit ODT, enter P (proceed). You must then issue a CTRL/D to return to the RSX system prompt.

Example 5–1: A Sample CCR Session

```
> CCR NODE DALLAS RET
CCR -- Remote console is reserved
 CTRL/B
002160
@$0/000000 LF
R1/005621 LF
R2/112000 LF
R3/000000 LF
R4/000001 LF
R5/000000 RET
@100/016122 LF
000102/000340 LF
000104/001614 LF
000106/000341 RET
@P RET
 CTRL/D
CCR -- Remote console is released
>
```

6

Choosing Network Buffer Parameters

DECnet-RSX software uses several types of buffers to perform network communications. The buffers are used to store messages being sent and received between DECnet nodes. The permanent database contains network parameters that determine the size, number, and memory allocation for these buffers. During network generation, either you or the NETGEN procedure itself determines initial values for these parameters. Subsequently, you can use CFE DEFINE SYSTEM commands to modify them as necessary. The network buffer parameters include

- The size and maximum number of large data buffers (LDBs)
- The maximum number of small data buffers (SDBs)
- The maximum number of communications control buffers (CCBs)
- The minimum number of receive data buffers (RDBs)

This chapter describes these parameters and provides guidelines for modifying them so that your node will operate as efficiently as possible.

Example 6–1: Excerpt from a Sample NETCFE.CMD File

```
.
.
.
DEFINE SYSTEM MAXIMUM CONTROL BUFFERS 6
DEFINE SYSTEM MAXIMUM LARGE BUFFERS 7
DEFINE SYSTEM MAXIMUM SMALL BUFFERS 11
DEFINE SYSTEM MINIMUM RECEIVE BUFFERS 2
DEFINE SYSTEM LARGE BUFFER SIZE 280
DEFINE EXECUTOR SEGMENT SIZE 262
.
.
.
```

6.1 The NETCFE.CMD Command File

The NETCFE.CMD command file can be used as input to CFE. The file contains a series of CFE commands that defines the permanent database (CETAB.MAC). Included in the file are commands that define the network parameters discussed in this chapter. NETCFE.CMD can be used as a template for making changes to the permanent database. In this case, revise a copy of the file and keep the original NETCFE.CMD for reference. Example 6-1, an excerpt from an NETCFE.CMD file, shows CFE commands that define network buffers.

The *DECnet-RSX Guide to Network Management Utilities* describes the CFE DEFINE SYSTEM command, which you use to modify network parameters in the permanent database.

6.2 Allocating Memory for Buffer Space

Deciding how much memory to dedicate to buffer space always involves trade-offs between minimizing memory usage and optimizing performance. For most users, the values allocated by NETGEN are adequate for establishing reasonable buffer resource requirements for support of the node being generated. The information provided here is for users who want to further optimize both memory utilization and DECnet performance. Table 6–1 provides the following information about each buffer type:

- How the buffers are used
- Where the buffers are located
- The size of the buffers

- Factors that determine the number of buffers required

All DECnet buffers, with the exception of communications control buffers (CCBs),are located in the network buffer pool. The default buffer pool is named POOL..and is created as a common partition. All DECnet processes must be mapped to communications control buffers. To facilitate this mapping, CCBs are allocated in executive address space.

Table 6-1: DECnet Buffers

Large Data Buffers	Small Data Buffers	Communications Control Buffers
Use		
All receive messages, user data transmissions, logical link connects	User interrupt messages, ECL link service messages, node talker messages	DDCMP control messages, DECnet interprocess communication, one for each active SDB and LDB
Location		
Network buffer pool, common partition POOL..	Common partition (POOL) in network buffer pool	Executive address space (DSR)
Size		
Variable (user specified, determines ECL segment size)	34 bytes	38 bytes
Factors Determining Number of Buffers Required		
Number of lines, device types, number of logical links simultaneously active and performing data transfers, transit traffic	Number of logical links simultaneously active and number of lines	Number of LDBs and SDBs

6.3 Large Data Buffers

DECnet-RSX software uses large data buffers (LDBs) for transmitting and for storage of all received user data. The buffers are called transmit buffers and receive buffers, respectively. DECnet-RSX software also uses LDBs on routing nodes for the storage of all route-through packets. These are also called receive buffers. During NETGEN, the user specifies the size of LDBs. NETGEN itself calculates the number of LDBs to be allocated based on this and other NETGEN specifications.

6.3.1 Determining LDB Size

This section discusses factors in determining the optimum LDB size for your node. The LDB size that you define at NETGEN determines the node's segment size, which is the size into which large user messages are divided for end-to-end transmission. NETGEN calculates the segment size to be *ldbsize*-18. After NETGEN, use CFE or NCP to change segment size (DEFINE or SET EXECUTOR SEGMENT BUFFER SIZE). If the segment sizes of communicating nodes—the nodes at either end of a logical link—differ, both nodes use the smaller of the two sizes. For communication to be successful, the chosen segment size must not be too large to be handled by intermediate nodes—that is, nodes that fall in the routing paths between source and destination. The LDB size for a routing node equals the maximum size of route-through packets that the node can handle. If all nodes within a network have the same LDB size, route-through packets too large to handle will not cause end-to-end communications to fail. In some cases it may be necessary to reduce the local node's segment size to ensure that transmitted packets can successfully traverse the network. Likewise, it may be necessary to increase the local LDB size to ensure that your node can handle route-through packets. Additional factors you must consider in defining LDB size include

- The record-blocking factors used by your application programs
- The amount of memory that you wish to allocate to LDBs
- The error rate of the communication lines
- Your throughput requirements

For example, if your applications are blocking data into 256-byte records and the LDB size is 256, DECnet software must split each record into two segments in order to transmit it. With an 18-byte End Communications layer (ECL) header affixed to each segment, one segment would have 238 bytes of data and an 18-byte header, and the other segment would have 18 bytes of data and an 18-byte header. In this type of situation, you would be incurring the overhead of segmenting records without using the segments efficiently, and at the same time you would be using up more memory than necessary for buffer

space. The obvious solution would be to increase the LDB size defined for the end nodes and thereby establish a larger segment size that could accommodate the 256-byte records without splitting them. But this would add to the amount of memory allocated to LDBs: something you might wish to avoid. As the size of the segments increases, the effective throughput rate will decrease. Effective throughput is the amount of data received without errors versus the amount of data received with errors. To resolve the situation in a suitable manner, you could establish an LDB size, and the ECL header size of 18 bytes, that is an integer multiple of the record size used by your application programs. In this way you would minimize the amount of memory allocated to LDBs and increase the throughput.

6.3.2 Number of LDBs to Allocate

Major factors that determine the number of LDBs you should allocate include

- The number and type of local devices and controllers
- The amount of route-through traffic anticipated if yours is a routing node
- The number of logical links likely to be active simultaneously

The last row in Table 6–1 summarizes these factors, which you can use to help determine how many LDBs you need to maintain a reasonable level of performance. For most nodes, some trial and error is generally required to arrive at an optimum number. Note that NETGEN itself calculates the minimum number of LDBs to be reserved for use as receive buffers; this number is called the minimum receive threshold. When the number of available LDBs is equal to or less than the threshold, only requests for receive buffers will be serviced. In this way, receive traffic gets priority over transmit traffic. Requests for transmit buffers will be queued until the number of available LDBs is above the threshold. The receive buffer threshold should be set no higher than the sum of the buffers required for all lines on the system. To redefine the minimum receive threshold, use the CFE DEFINE SYSTEM MINIMUM RECEIVE BUFFERS command. Table 6–2 gives the recommended number of LDBs that should be reserved as receive buffers for each communications device driver supported by DECnet-RSX and RSX–11 PSI.

Table 6–2: LDBs Required by Different Device Types

Device Type	Number of LDBs
DL,DLV	1
DUP,DU,DUV	1
DV	1 per line
DZ,DZV	1 per line
DHU,DHV	1 per line
KDZ,KDP	1 per line
KMY,KMX	6 per line
KMV	2 per line
PCL	1
DMC (low speed, up to 19.2K bits/second)	2
DMC (high speed, 56K to 1M bits/second)	4
UNA,QNA	6

Routing nodes require more buffers than nonrouting nodes to provide space for route-through traffic. If there are insufficient LDBs, route-through packets will be discarded, resulting in degraded end-to-end performance. The following algorithm can be used to calculate the number of buffers required for a particular configuration to support route-through traffic. It assumes that route-through traffic contributes less than 60 percent of the total bandwidth of all lines and will give a low probability of packets being discarded.

$$number - of - buffers = 5 * \sqrt{n}$$

where n is the number of lines.

Nonrouting nodes do not require buffers to support the route-through capability. Nonrouting nodes can be configured with more large buffers than required for the device without affecting node performance. Table 6–2 indicates the number of LDBs that should be reserved for receives on nonrouting nodes. In this case, the receive buffer threshold should be set to the value required for the device or 2, whichever is higher.

6.4 Small Data Buffers

DECnet-RSX software uses small buffers (SDBs) to send user interrupt messages, End Communications layer (ECL) link service messages, and some control messages. SDBs have a fixed length and are used in place of LDBs in order to optimize resource utilization. NETGEN calculates the number of SDBs to be allocated. At least one SDB is required for each of the logical links that are in use simultaneously. An additional SDB should be allocated for each physical link. NETGEN calculates a default number. Then, when the system is in operation, you can use the Network Display Program (NTD), described in the *DECnet-RSX Guide to Network Management Utilities* , to monitor the number actually in use. If necessary, you can adjust the number of SDBs allocated using the CFE DEFINE SYSTEM MAXIMUM SMALL BUFFERS command.

6.5 Communications Control Buffers

Each active LDB and SDB, while it is in use, needs to have a communications control buffer (CCB) associated with it. DECnet-RSX software also uses CCBs to pass parameters between processes and to send DDCMP control messages. CCBs are fixed length. NETGEN calculates the minimum number of CCBs to be allocated. If more are needed, they are allocated dynamically. One CCB is required for each active LDB and SDB. One CCB is generally required for each logical link. An additional CCB should be allocated for each of the lines that will simultaneously be in the ON-STARTING state, that is, lines set to the ON state but not connected to an adjacent node. NETGEN calculates this value. Use NTD to monitor the number of active logical links and to detect allocation failures. You have reached the optimum number when an allocation failure occurs only occasionally.

7

Area Routing Guidelines

DECnet-RSX now features an enhanced routing capability that permits you to configure network nodes into groups called areas. These areas are then linked together to form a complete network. This capability, called area routing, represents a two-level, hierarchical approach to network configurations. This chapter presents recommendations and guidelines for configuring networks that use area routing and discusses problems that can occur if a multiple area network is not configured correctly.

7.1 Summary of Phase IV Node Types

The concept of area routing was introduced in Section 2.2. In that section is a discussion of node types supported by DECnet Phase IV. These node types are summarized in the following list:

- A **Phase IV level 2 router** keeps information on the least cost path to all other areas in the network and to all other nodes within its own area.
- A **Phase IV level 1 router** keeps information on the least cost path to all nodes within its area including its nearest level 2 router. All packets whose destination is outside the local area are forwarded to the nearest level 2 router.
- A **Phase IV end node** is a single-circuit node that keeps no information about least cost paths. All outgoing packets are sent to the single router, level 1 or level 2, to which the end node is connected. If the node is on an Ethernet, direct communication between end nodes on the Ethernet is allowed.

- A **Phase III router** keeps information on the least cost path to a maximum of 255 nodes within its own area. It does not keep information about its nearest level 2 router and does not support multiple area routing. A Phase III router cannot be used to route packets outside its own area and cannot route packets to nodes whose node number is above 255. Phase III routers do not include Ethernet support.
- A **Phase III end node** is a single-line node that keeps no information about least cost paths. All outgoing packets are sent to the single router, level 1 or level 2, to which the end node is connected. A Phase III end node cannot be used to send packets outside its own area and cannot send packets to nodes whose node number is above 255. Phase III end nodes do not include Ethernet support.

7.2 Area Routing Guidelines

Configuring a multiple area network is more complex than configuring a single area network. The design of a multiple area network creates certain network topological restrictions and design considerations that were not present in single area networks. The area routing guidelines presented here are based on these restrictions. The guidelines are intended to prevent problems such as loss of routing paths, isolation of nodes, and incorrect routing of packets. These potential problems are discussed in Section 7.5.

When you configure a multiple area network you should follow these guidelines:

- **Every level 1 routing node must be in only one area.** This restriction states that it is illegal for a level 1 router to have a circuit to a node outside its local area.
- **Each area, level 1, network must be physically intact.** All nodes within an area must be connected in some way to all other nodes within the same area. The nodes do not need to be physically adjacent; however, a physical path that lies totally within the local level 1 area must exist. This path can involve level 2 routing nodes in the same area.
- **The level 2 network must be physically intact.** All level 2 routing nodes must be connected in some way to all other level 2 nodes. This connection must only involve level 2 nodes. Level 1 nodes or Phase III nodes cannot form a part of the path between two level 2 routing nodes.

- **Areas should be used to reflect expected traffic flow.** This guideline, while not mandatory, will greatly improve overall network performance. Routing within an area is less costly; therefore, areas should be set up to take advantage of this fact. When setting up an area, try to include in the area all nodes that perform some logical function. Areas should reflect not only physical node location but a combination of both physical and logical node location.
- **Areas should not be used to enforce protection.** The DNA architecture does not specify protection guarantees for areas, and no assumption about such guarantees should be made when setting up areas. Areas were not designed to be used as an accounting or security tool.

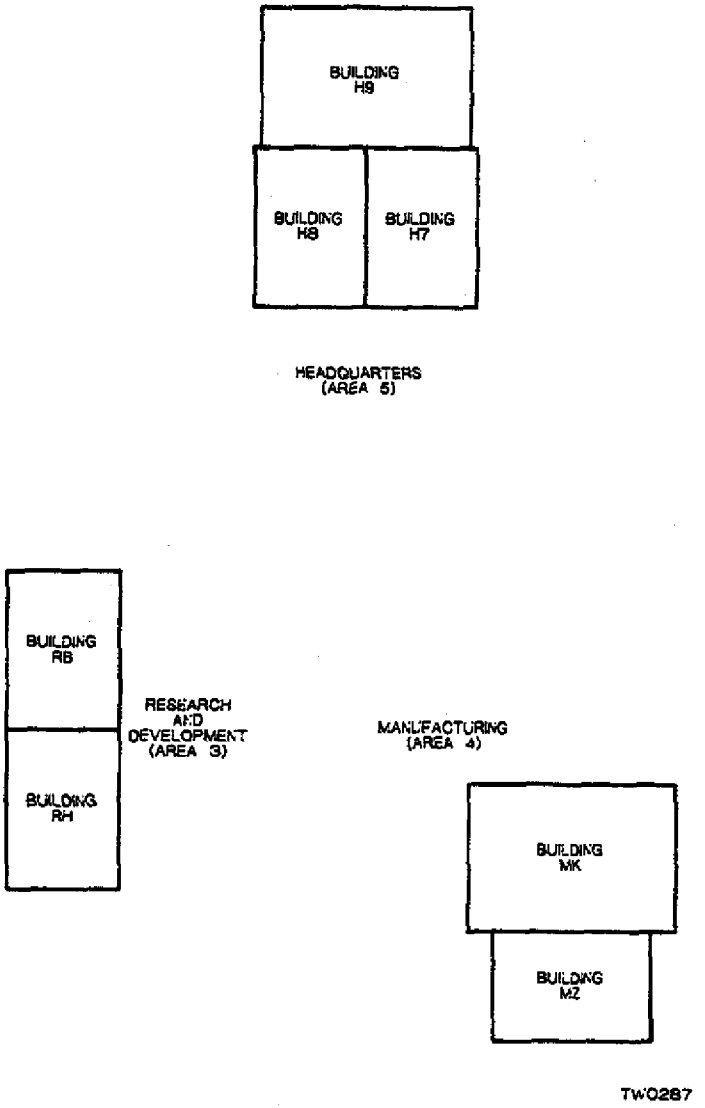
The next section offers an example of the design process that you should follow when designing a multiple area network.

7.3 Designing a Multiple Area Network

This section illustrates the use of the guidelines in designing a multiple area network. The goal of the design process is to build a robust, redundant network that will not be subject to a single point of failure.

Figure 7–1 shows the building layout of a hypothetical company. These buildings are located in four separate cities and perform three different functions. The four locations are some distance from each other and leased phone lines will be used for communication between locations. This building layout lends itself very well to the concept of areas. Except for the manufacturing, traffic within each location is expected to be greater than traffic between the locations. In the case of manufacturing, the two separate locations are included in the same area.

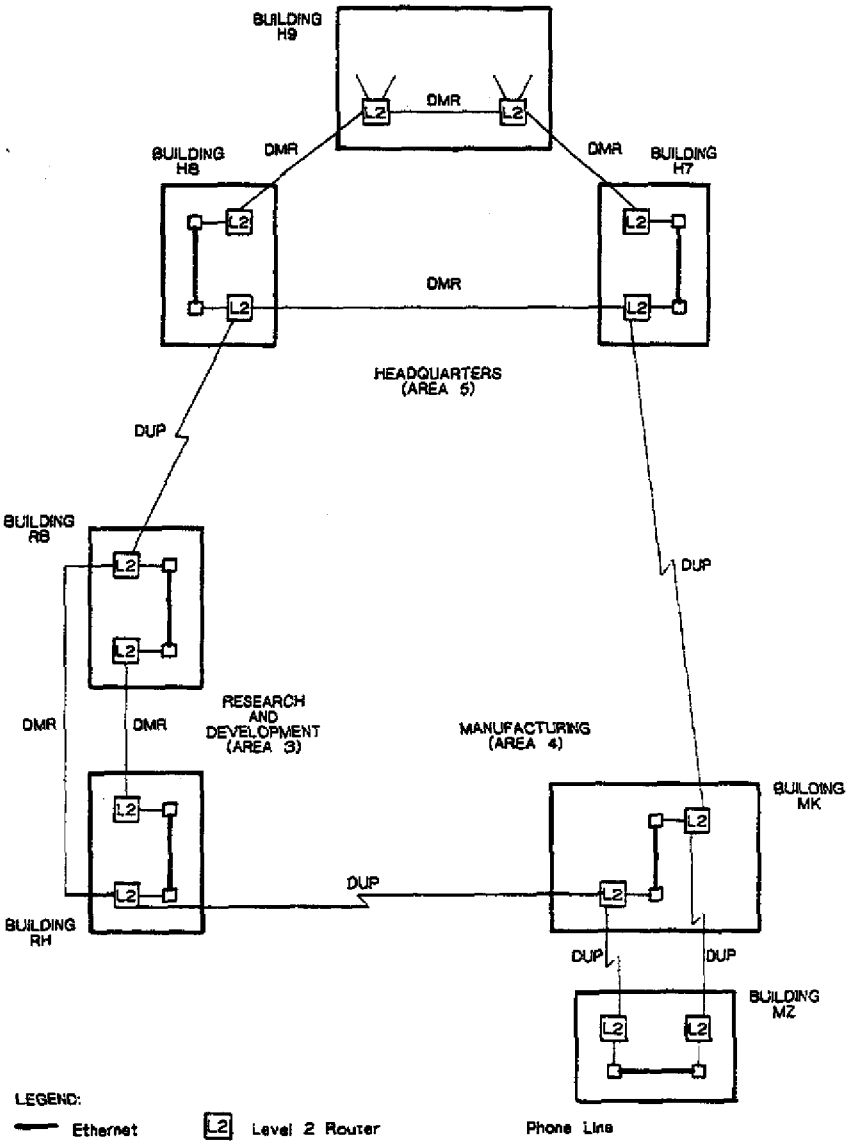
Figure 7–1: A Typical Company Requiring a Computer Network



The level 2 network designed for this application is shown in Figure 7–2. This network offers the following strengths:

- Each level 2 node is redundant. No single node failure will affect the level 2 network. For example in area 5, if one of the level 2 routers in building H9 fails, all level 2 routers in area 5 can still communicate with each other and with the level 2 routers in the other areas.
- Each level 2 line is a redundant path. No single physical line failure will affect the level 2 network. For example, if the DUP line between area 5 and area 3 fails, all level 2 routers can still communicate with one another.
- The areas have been designed to reflect traffic flow. The maximum amount of traffic is within each area. If one section of the network experiences heavy traffic flow, only the users of that area are affected.

Figure 7–2: The Level 2 Network

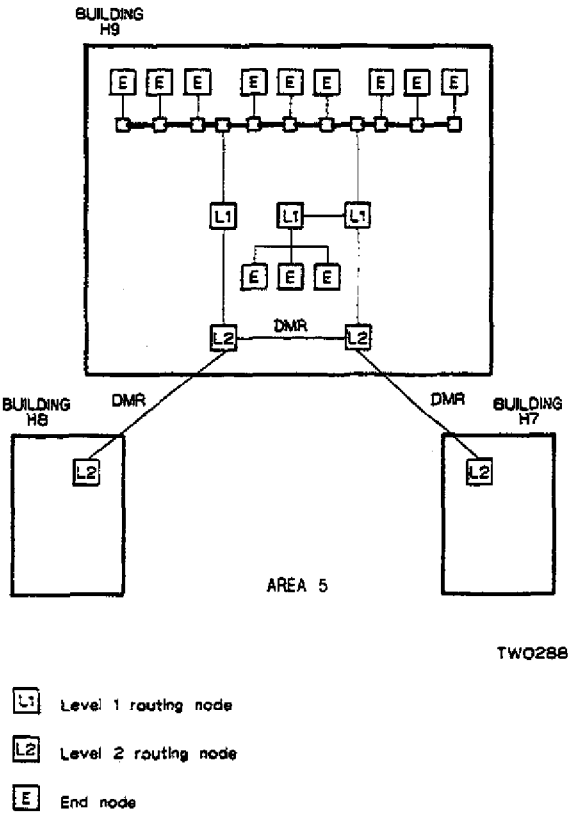


Each area is a complete level 1 network. When configuring each area, you must ensure that there are redundant paths to the level 2 routers in the area. Redundant paths help to maintain the availability of the other areas in the network.

Figure 7–3, which shows the headquarters portion of the level 1 network, illustrates redundant paths with a circular topology. An Ethernet has been added in building H9. This Ethernet has two level 1 routers, each of which is connected to one of the level 2 routers. This topology provides an alternative path for the end nodes on the Ethernet if one of the level 1 routers should fail. This topology allows the Ethernet end nodes to reach other network nodes even if a single failure occurs. In some cases, as in the level 1 router shown with the multipoint line attached, the redundancy from circular networks cannot be used due to other topological restrictions.

Node performance in outside areas will not be affected by your adding nodes in a given area. The maximum number of nodes for an area is 1023. The concept of areas should be used to balance the routing overhead load among all level 1 and level 2 routing nodes. Therefore, you should always strive to have areas that are roughly equal in size.

Figure 7–3: A Portion of the Level 1 Network in Area 5



Once you have designed the network from the level 2 routers down to the end nodes, you should actually install the network in a bottom-up manner:

1. Test each network node using the installation checkout procedures for that node. For DECnet-RSX nodes, see the postinstallation checkout procedures described in the *DECnet-RSX Network Generation and Installation Guide*.
2. Starting with the end nodes, connect all the nodes within each area.
3. Fully test each area.
4. Connect all the areas.
5. Fully test the entire network.

7.4 Converting to a Multiple Area Network

Converting an existing network into a multiple area network requires careful planning. Because you will be changing network node addresses, there may be a period during the conversion when some nodes will be unreachable. The following steps provide a course to follow which will keep the transition disruption to a minimum.

1. Plan ahead. Using the guidelines discussed in Section 7.2, completely define what the entire network topology will be after the conversion. Decide which nodes will be level 2 routers, level 1 routers, and end nodes.
2. If the new design requires that some current nodes be repositioned, make the required changes before you begin the conversion. For example, you may decide that a heavily used node currently in the place where a level 2 router will be cannot handle the additional routing load associated with a level 2 router.
3. Create a new permanent database for each node. On DECnet-RSX nodes, you must perform a network generation to include area support. Some implementations of DECnet, such as DECnet–VAX, do not require a generation, but all nodes will require a redefining of the DECnet operating parameters. In this step you will also have to define all the network nodes in the database using their new multiple area addresses. If the change requires only database changes, it is highly recommended that you perform this step using a copy of the permanent database and not the actual permanent database or the running database. In this way, you do not disrupt the running network and you have the capability of restoring the current running network in the event of a system failure. On DECnet-RSX nodes, which require a network generation, this means you should do the network generation in a UIC other than the current network UIC.

NOTE

Do not forget to change your down-line load and up-line dump database to reflect the new node addresses.

4. After you have created new permanent databases for all the network nodes, make these databases the actual permanent databases by moving them to the running network account or UIC. On DECnet-RSX nodes, you should redefine the network UIC.

5. Shut down the running network and bring it up again with the new permanent databases. This step benefits from some real-time cooperation among the node managers in the network. If this cooperation is not feasible, then avoid using area number 1 for any area in the new network since current implementations of DECnet use area 1 in a single area network.

   At approximately the same time, have each node manager shut down DECnet on the node. When all nodes have shut down DECnet, or after an agreed upon time interval, bring up the network using the new network database.

   Note that if your node is on an Ethernet and is supporting applications other than DECnet, such as LAT, these applications should be shut down along with DECnet. This is necessary because the Ethernet physical address of your node will be changed to reflect your new node address. After DECnet is running, you can restart any other network applications.

6. There may be a period of debugging the network to ensure that all desired connections have been made. You can simplify debugging the conversion if you can run each area independently for a short time before bringing the area into the network. You can do this by setting all level 2 circuits to the OFF state in the new copy of the database. Once you are confident that an area is operating correctly, turn these circuits on to link up adjoining areas.

7.5 Problems in Configuring a Multiple Area Network

The use of area routing can lead to certain network problems that may not be readily identifiable. This section describes some of the problems related to violation of the area configuration guidelines presented in Section 7.2 and explains how to solve these problems.

7.5.1 Partitioned Area Problem

Improper configuration of a multiple area network can result in a failure condition known as the partitioned area problem. This failure condition occurs when a line or node failure breaks an area into two or more separate pieces, known as partitions, with each partition still attached to the rest of the network. An example of this problem is shown in Figure 7–4.

In this network the area shown has been improperly designed; the area is highly vulnerable to a single point of failure (line W). An explanation of what happens when line W goes down will show how the partitioned area problem occurs and what the result of this problem is.

Assume all circuit costs in Figure 7–4 are equal to 1. Node D in area 8 attempts to communicate with node A in area 5. If line X or Z is down, or if node C or F is unavailable, there is no problem; the routing algorithms will route packets from D to A over the remaining path. However if line W goes down, there will be a problem due to the following routing decisions:

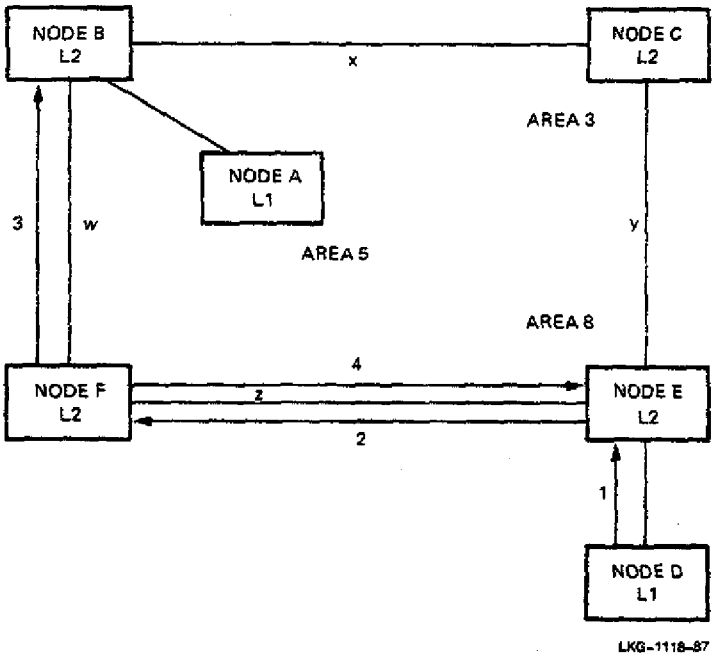
1. Node D, using level 1 routing, sends the data packet to node E, its nearest level 2 router, in order to get the packet outside its area.
2. Node E, using level 2 routing, routes the packet to node F because this level 2 router is on the least costly path to area 5.
3. Node F, using level 1 routing, tries to send the packet to node B. However, because line W is down, the attempt fails.
4. The packet is returned to the sender marked undeliverable. Node D sees node A as being unreachable even though there is a physical connection, through node C, between the two nodes.

The problem is further compounded because, even if line W is down, node A can communicate with node D. This is because node B, using level 2 routing, sends the packet through node C to node E which then passes it along to node D. In this situation node D cannot communicate with node A; however node A can communicate with node D. Therefore a "one-way" link is created.

The point to remember from this example is that level 2 routing and level 1 routing are independent of one another. The source node uses level 1 routing to send the packet to its nearest level 2 router. The level 2 router uses level 2 routing to pick the least costly path to the destination area. The level 2 router in the destination area then uses level 1 routing to send the packet to the destination node within its area. Therefore, any time you have a situation where an area is reachable, but a node within the area is not reachable from all level 2 routers in that area, you have a potential partitioned area problem.

Area partitioning is a problem that can be avoided by implementing a good network design. The only remedy for the problem of partitioning is to fix the broken line or node. Designing an area as a straight line network, as in area 5 (Figure 7–4) should be avoided. To prevent the problem shown in the example, add a second line between nodes B and F to provide an alternative path should line W go down.

Figure 7–4: A Network with a Partitioned Area Problem



7.5.2 Phase III/Phase IV Integration Problems

In a Phase IV multiple area network, Phase III nodes can be included with some important restrictions:

- Phase III end nodes do not support Ethernet.
- All Phase III nodes, both routing nodes and end nodes, can communicate only with nodes within the same area as themselves.
- All Phase III routing nodes can communicate only with nodes for which they can maintain routing information. Phase IV DECnet nodes can have higher node numbers than older Phase III nodes can address because Phase III's node number limit was 255. For this reason, any Phase IV node with an address above 255 is "invisible" to all Phase III nodes in the area. This is a good reason to break a network into areas that do not exceed 255 nodes even if the limit of 1023 nodes has not been reached.
- A Phase III node cannot be in the path of two Phase IV nodes with node numbers higher than 255.
- A Phase III node must not be in the path of any inter-area communication. Because Phase III nodes do not know about areas, they will not handle inter-area packets correctly.
- Phase III nodes must use routing initialization passwords when they are connected to Phase IV nodes. You can specify, in your local configuration database, transmit and receive passwords for each adjacent node. The transmit password is that which you send to the remote node. The receive password is that which you expect to receive from the SET NODE command to specify these passwords. Each password can be one to eight alphanumeric characters long.

The formats for Phase III and Phase IV node addresses are not the same. Phase III node addresses consist of a node number in the range of 1 to 255. Phase IV node addresses consist of an area number and a node number. For more information about the node address, see Section 2.2.

Whenever a Phase III node is brought up in a Phase IV network, the physical link between the nodes is initialized using Phase III protocol. This affects the packet headers in the following ways:

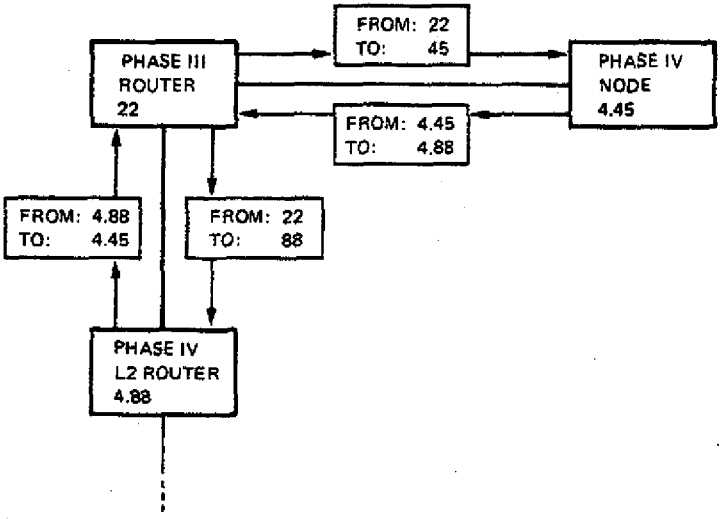
- **Phase IV → Phase III traffic:** The area number is discarded by the Phase IV node and replaced by a zero. For example in Figure 7–5 when node address 4.88 is passed to a Phase III node, the node address becomes 88.

- **Phase III → Phase IV traffic:** The Phase IV node adds the local area number to the node address. For example in Figure 7–5, if node address 45 is sent to Phase IV node 4.45, the node address 45 becomes 4.45.

These different address formats are responsible for two types of configuration problems that lead to lost packets, lost packet acknowledgments, or incorrectly routed packets:

- The problem of incorrectly placing a Phase III node in a Phase IV routing path.
- The problem of placing a Phase III node between two areas which creates an area "leakage" problem.

Figure 7–5: A Phase III Routing Node in a Phase IV Routing Path



7.5.2.1 A Phase III Routing Node in a Phase IV Routing Path

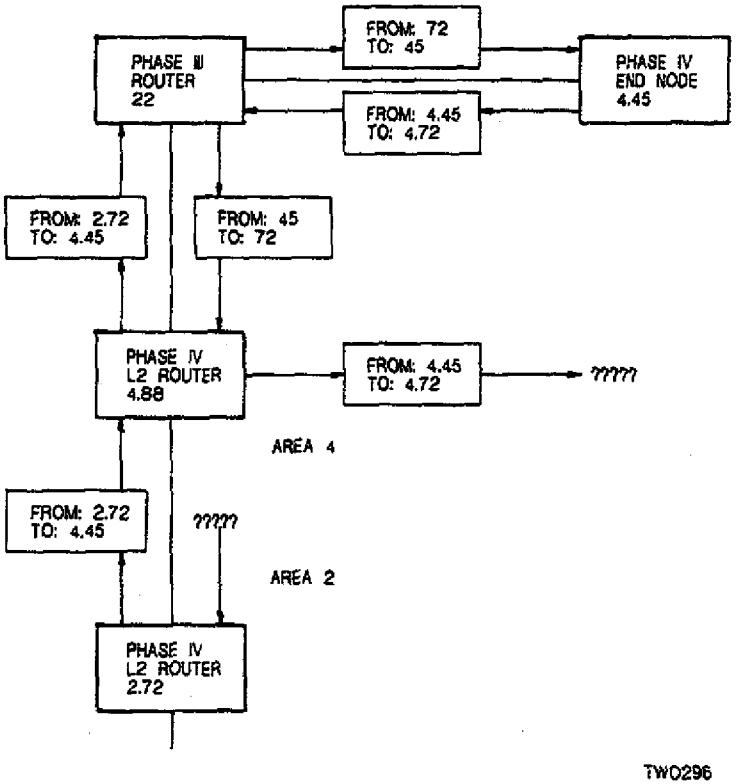
No problem occurs with a Phase III node in a Phase IV network as long as none of the nodes in the Phase III node's area have addresses over 255 and all logical links going through the Phase III node are for communication paths entirely within the local area. In the example shown in Figure 7–5, if node 4.88 sends a packet to node 4.45, the packet first goes to the Phase III node 22. Node 4.88 discards the area number 4 from the source and destination address in the packet before sending the packet to node 22. Node 22 sends the packet to node 45. Note that no area number is used. Node 4.45 adds its area number to the source and destination address of the packet, making the addresses 4.88 and 4.45 respectively. During the return trip the same procedure is followed.

A problem occurs when a Phase III node is between two Phase IV nodes that form an inter-area communications link. For example, if node 2.72 in Figure 7–6 sends a packet to node 4.45, the packet first goes to node 4.88, and then to the Phase III node 22. When node 4.88 sends the packet to node 22, node 4.88 drops the area numbers from the source and destination address. When node 22 passes the packet to node 4.45, the Phase IV node adds the area number (4) to the source and destination address. This makes the source address 4.72, which causes the return packet to be routed to node 4.72 instead of 2.72. If node 4.72 exists, the packet is incorrectly sent to that node; node 2.72 will eventually time out waiting for its reply. If node 4.72 does not exist, the reply packet will be returned to node 4.45 marked undeliverable.

If a Phase III node is placed between two high address Phase IV nodes, the Phase III node will not be able to keep routing information for these nodes and will be unable to forward packets between the nodes. For example, if a Phase III node is placed between nodes 4.12 and 4.589, any packets sent from node 4.12 to node 4.589 will be lost because the Phase III node cannot keep routing information for node number 589.

To avoid the problems just discussed, you should always place Phase III nodes on the periphery of the network. In this way Phase III users notice no change and Phase IV users are not restricted to communicating only with nodes in their area.

Figure 7–6: An Example of an Improperly Placed Phase III Node



7.5.2.2 Area Leakage Problem

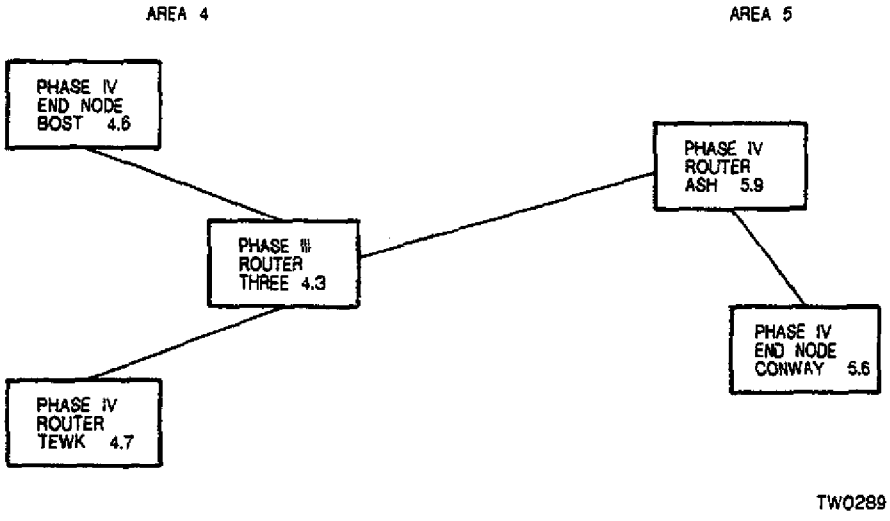
Area leakage is the leaking of level 1 routing information between two areas. When this occurs, a Phase IV node in one area can communicate with a Phase IV node in another area without having to use level 2 routing. This problem is caused by improperly placing a Phase III node in a Phase IV multiple area network. If a Phase III node is directly linked to two areas it will build up a routing database containing routing information for both the areas. The Phase III node will then transmit this routing information to Phase IV nodes in both areas. This allows two Phase IV nodes, each of which is in a different area, to communicate without having to use level 2 routing.

Area leakage can be prevented by requiring all Phase III nodes in the network to use routing initialization passwords. When you change from a single area Phase III area network to a multiple-area Phase IV network you should define new passwords for all Phase III nodes that will be members of the new network. The passwords you use should have the area number encoded in them. Phase IV nodes always require a password from a Phase III node. By setting all new passwords in the manner described, you will be able to locate all Phase III nodes that are trying to connect across areas.

Note that this technique will only work if you define new passwords for all Phase III nodes in the manner described.

Figure 7–7 shows a Phase III node that is causing area leakage between areas 4 and 5. The following sequence, based on Figure 7–7, shows how this occurs and why this can be a problem.

Figure 7–7: Area Leakage with Phase III Nodes



1. Node THREE is a Phase III node that is a member of areas 4 and 5. It builds a routing database consisting of nodes in areas 4 and 5. Because node THREE is a Phase III node, none of the entries in the database has an area number associated with it. This database is shipped to all routing nodes in areas 4 and 5.

2. Node ASH will assume that the routing information it receives from node THREE is only for nodes within ASH's area. Because node ASH is a Phase IV node, it will add its area number (5) to the database entries before using the routing information. Therefore, node ASH will think nodes BOST and TEWK have addresses 5.6 and 5.7 when they really are nodes 4.6.and 4.7.

3. A problem occurs when a node of the same number as one of the "leaked" nodes already exists within the area. In the example, this happens with node CONWAY. To node ASH, CONWAY is node 5.6; when it receives the routing database from node THREE, there will be two nodes with the same address defined. As in Phase III networks, the node to which ASH will send packets for node 5.6 will depend on the cost of the path to each of the two 5.6 nodes. If the circuit cost to CONWAY is high, the packets will be sent to node BOST. If all circuit costs are 1, then the packets would go to node CONWAY.

The previous example shows one of the two problems of area leakage; area leakage also occurs even if two nodes do not end up with the same address. For example, assume node CONWAY in Figure 7–7 has address 5.4. If the first two steps described here occur and node CONWAY wanted to talk to node BOST, it would send the packet to node 4.6. However, due to the effect of step 2, the packet would be returned marked undeliverable. This is because node ASH only has an entry for 5.6.

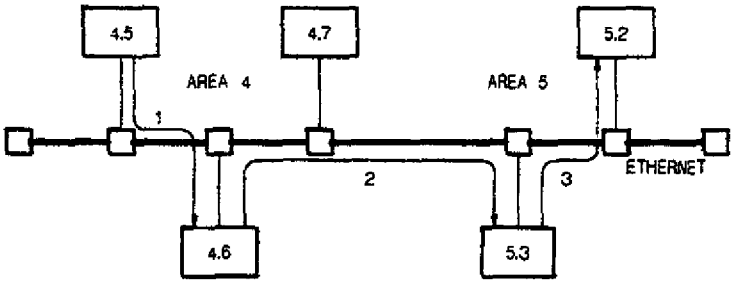
7.6 Multiple Area Networks and Ethernet

Avoid setting up more than one area on a single Ethernet. Phase IV DECnet supports such configurations, but a multiple area causes a large increase in traffic on the Ethernet. This traffic increase occurs because a packet must be routed through two level 2 routers instead of being sent directly between two nodes. Figure 7–8 shows an Ethernet with two areas. In this example, if node 4.5 wants to send a packet to node 5.2 the following sequence takes place.

1. Node 4.5 sends the packet to node 4.6, using level 1 routing to its designated router.
2. Node 4.6 sends the packet to node 5.3, using level 2 routing to the destination area.
3. Node 5.3 sends the packet to node 5.2, using level 1 routing to the destination node.

As this example shows, three packet transmissions take place where only one would occur in a single area Ethernet.

Figure 7–8: A Multiple Area Ethernet



8

A Component View of DECnet-RSX/PSI

DECnet-RSX operation is based on the interaction of a large number of components. This chapter provides a comprehensive overview of the way these components work together in functioning as a DECnet-RSX node and is intended for a system manager or an advanced user of the system. Components include:

- Communications Executive components
- DECnet communications components
- CEX support components
- System management utilities
- System management support components
- DECnet utilities
- Satellite support components
- PSI communications components
- PSI support components

8.1 The Communications Executive

The Communications Executive (CEX) coordinates the scheduling and running of processes. These processes are memory-resident modules of code which perform specific steps in message processing. These processes are described here.

8.1.1 Three Types of CEX Processes

CEX schedules and monitors three types of processes: logical link control (LLC) processes, data link control (DLC) processes, and device driver module (DDM) processes. CEX provides an environment that separates the functions of each process.

- **Logical link control (LLC)** processes provide end-to-end data transmission through establishing and maintaining logical links. An LLC ensures accurate transmission between the communicating nodes. The End Communications layer process and driver (ECL) and the Routing layer process (XPT) are LLCs that handle logical links and routing.
- **Data link control (DLC)** processes maintain a protocol that is designed to create an error-free transmission path between the two communicating physical devices. The Digital Data Communications Message Protocol (DDCMP) is a DLC process that handles transmissions or retransmissions if necessary. The Ethernet Protocol Manager (EPM) process is also a DLC process.
- **Device driver module (DDM)** processes control specific communications devices such as a UNA or a DUP device. There is one DDM process for each specific device type. On transmission, a DDM takes a block of data and executes the instructions required for the device to send the data. On reception, the DDM handles interrupts for received data and assembles an entire block of data to be passed on to the next process.

The Auxiliary Process

Logically connected to CEX is the auxiliary (AUX) process. AUX assists CEX in performing communications functions by providing the following subroutines:

- Buffer recovery
- Powerfail recovery
- Modem control (optional)
- Timer service
- Scheduling and monitoring of processes

CEX and AUX function as one unit, even though AUX is split out from CEX to maximize available system pool space. The amount of space allocated for CEX depends on the type of RSX operating system that is running DECnet-RSX.

For RSX–11M–PLUS and Micro/RSX systems, SYSGEN generates the Communications Executive within the RSX Executive. For RSX–11M and RSX–11S systems, the Communications Executive resides in its own partition called CEXPAR. For RSX–11M and RSX–11S systems, you add the partition CEXPAR to the system image file using virtual monitor routine (VMR) commands.

Partition layouts for all these RSX systems are described in the *DECnet-RSX Network Generation and Installation Guide.*

8.1.2 The CEX Environment

Transmission of data proceeds from a user program to a physical device. Figure 8–1 displays the LLC, DLC, and DDM processes within the CEX environment. The dotted lines symbolize the interface between each of the processes.

8.1.2.1 Transmitting Data

When a local user program is transmitting data to a user program on another node, the local user program issues a QIO call to the RSX Executive. In MACRO–11, the local user program issues QIO calls directly. In higher level languages, the local user program issues subroutine calls that are converted to QIO calls by DECnet library routines. These calls are described in the *DECnet-RSX Programmer's Reference Manual.*

The QIO call passes the address and size of the buffer containing the data and also specifies the driver that will handle the data transmission. The driver is an RSX pseudodevice such as NS: or NX:. When the RSX Executive receives a QIO call to a pseudodevice, it calls the driver associated with that pseudodevice, which, in this case, happens to be an LLC. For example, ECL is an LLC associated with the pseudodevice NS:. A QIO call to NS: will always cause the RSX Executive to call ECL.

LLCs serve as bridges between the QIO environment known to the RSX Executive and the process environment known to CEX. To the RSX Executive, they appear as a normal RSX driver, and to CEX they appear as an LLC process. During transmission, the LLCs are activated by QIOs through the RSX Executive. LLCs then call another LLC or a DLC (XPT -> DLC or DLX -> DLC). During reception, LLCs get called by CEX and then signal a status return to the task through the QIO mechanism. The sequence of processing data for transmission is:

1. A user program sends a QIO to the pseudodevice; the RSX Executive calls a driver (an LLC) to process the data. This LLC processes the data (ECL handling logical links and XPT handling routing) and adds the appropriate headers to the data being transmitted. The LLC then sends a transmit

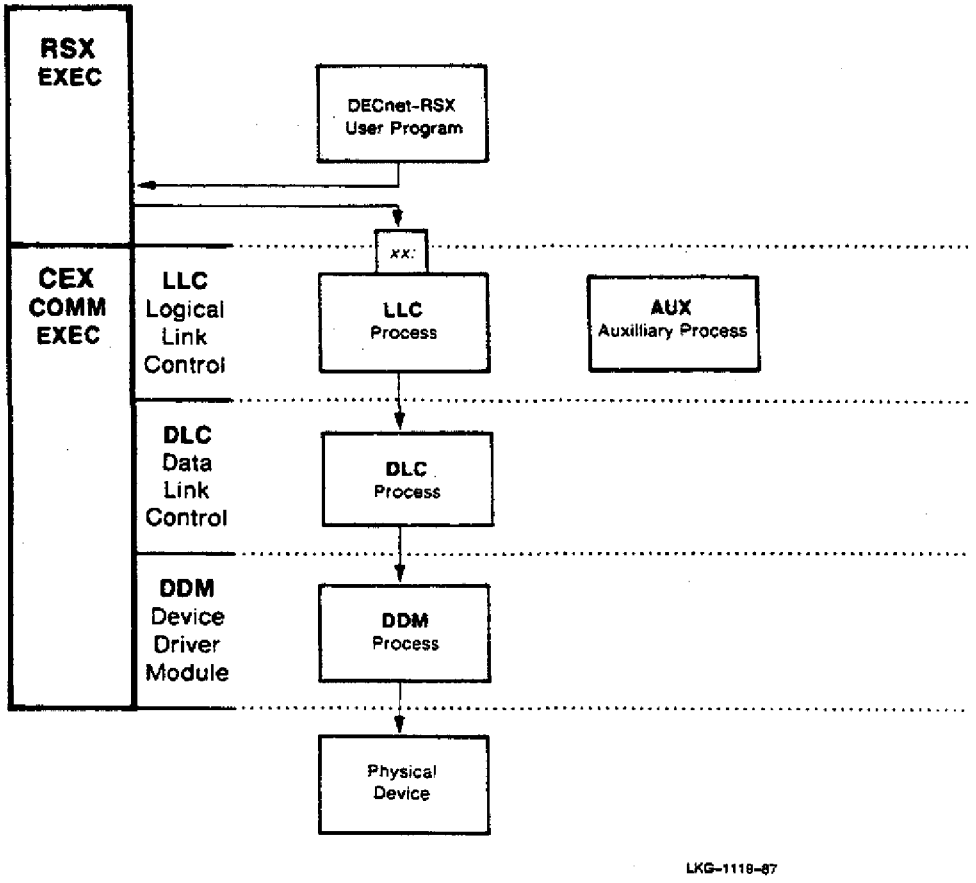
enable message to CEX to indicate it has completed its portion of the processing.

2. CEX then calls the next process, another LLC or a DLC. DLC processing ensures the data transmits accurately, and if necessary, retransmits the data. DLC adds its header to the headers and data supplied by the LLC. DLC then sends a transmit enable message to CEX to indicate it has completed its portion of the processing.

3. CEX then calls the appropriate DDM to send the data out to the physical device. In this series of exchanges, data is not actually moved, but rather pointers to it are transferred from process to process. The DDM controls the physical device through hardware I/O instructions. So there is a direct connection between the DDM and the physical device being driven. Finally, the physical device transmits the data to the line to the remote node.

4. When the DDM has completed transmission, the DDM then sends a transmit complete message to CEX. This message ripples back through the processes to the LLC, informing it that transmission terminated satisfactorily or not. If an error occurs, error status information is returned.

Figure 8–1: CEX Environment Illustrating Transmission

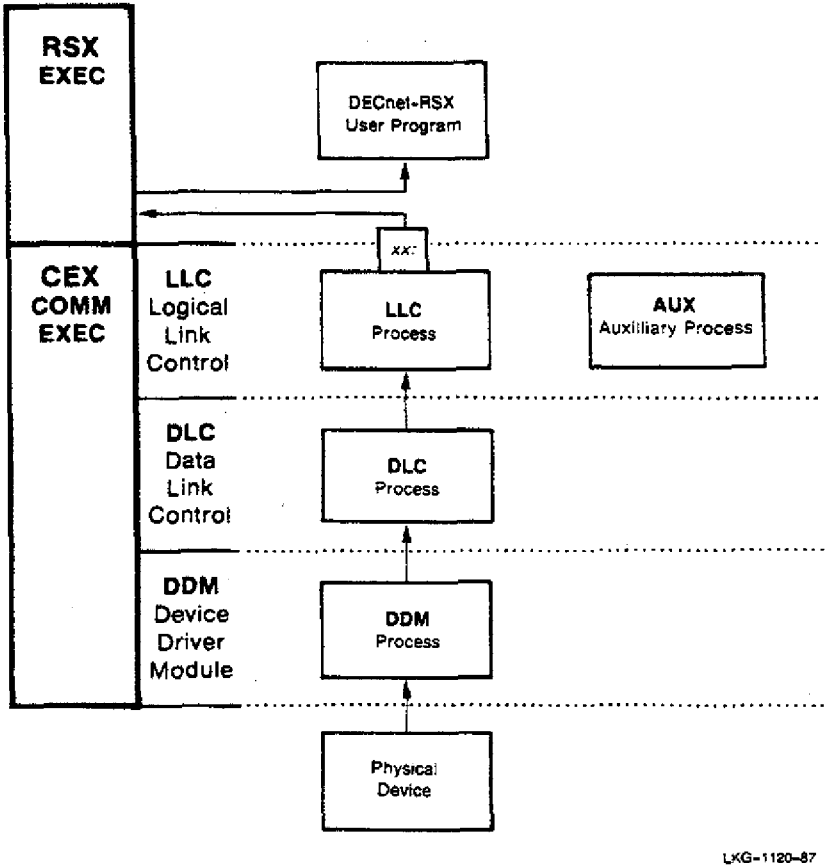


8.1.2.2 Receiving Data

The reception of data proceeds from a physical device to a user program. Figure 8–2 illustrates this process. Prior to receiving data, the user program on the receiving node must issue a QIO call indicating it wants to receive data. The sequence of processing data for reception is:

1. Upon receipt of the data, the physical device generates an interrupt message. The DDM associated with the particular device services the interrupt and assembles the data from the device until the entire message is complete. For a character-interrupt device, it will take a number of interrupts before the data block is completely assembled. The DDM then sends a receive complete message to CEX when the entire message has been received.
2. CEX then calls a DLC to process the data. The DLC checks the data's integrity and, if necessary, requests retransmission. The DLC removes the header supplied by the transmitting DLC. The DLC then sends a receive complete message to CEX when the entire message has been received. CEX then calls the appropriate LLC process.
3. The LLCs process the data and control the handling of logical links (ECL) and routing (XPT). The LLC then strips off the XPT and ECL headers, leaving only the original data. The RSX Executive then makes a status return to the original user QIO requesting data. This status return tells the user program it has received data from another node, and the user program then proceeds to process the data.

Figure 8–2: CEX Environment Illustrating Reception



8.2 DECnet Communications Components

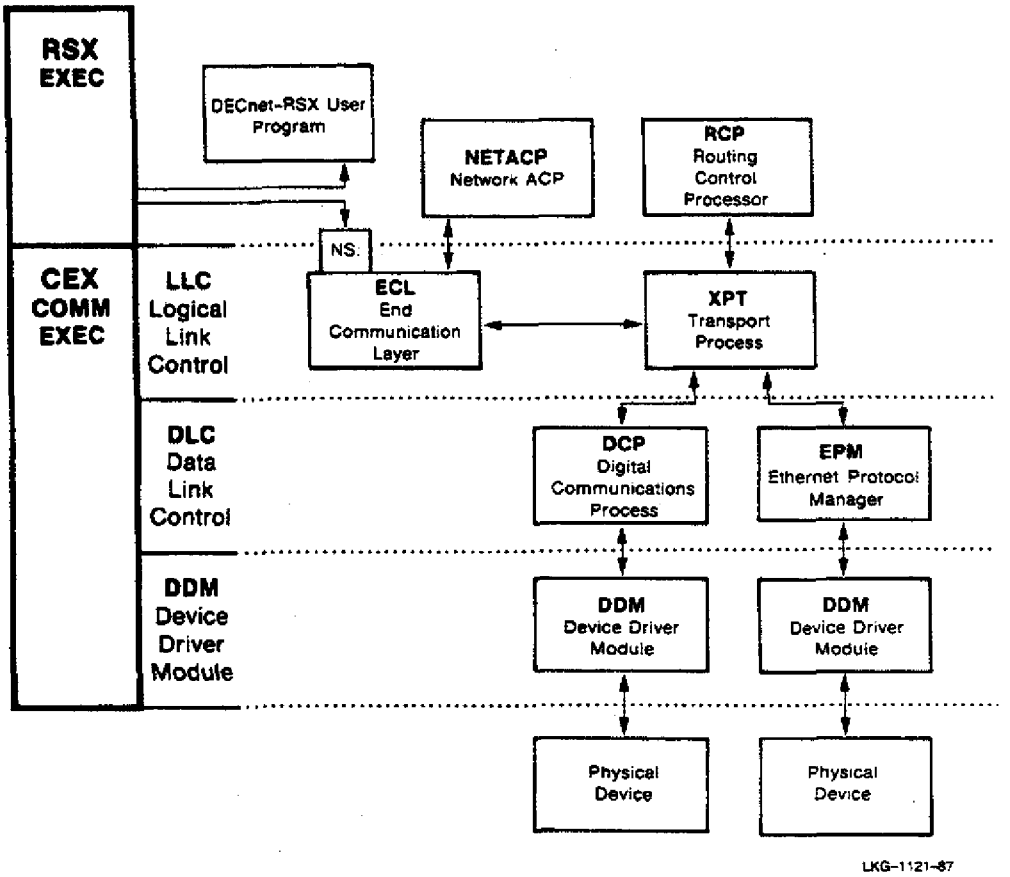
This section describes how components function within a basic DECnet environment and how the best routing pattern for messages is determined.

8.2.1 A Basic DECnet-RSX System

Figure 8–3 illustrates a basic DECnet-RSX system. The LLCs are ECL, the End Communications layer process; and XPT, the Routing layer process. The DLC can be DCP, the Digital Communications process or EPM, the Ethernet Protocol Manager process. DCP maintains DDCMP, the Digital Data Communications Message Protocol. EPM manages the Ethernet protocols. DCP can drive any of several different physical devices such as DUP, DZ, and KDP. EPM drives the physical devices (DEUNA, DEQNA, DELUA, and DELQA) using the Ethernet. The sequence of processing is:

1. If a DECnet user program sends a QIO call to the RSX Executive, the Executive passes the data to an LLC. In this case, the LLC is ECL. This LLC is known as the NS: pseudodevice to the Executive.
2. ECL processes the data and handles all network services. ECL calls the network ancillary control processor (NETACP) into memory to perform the less time-critical aspects of processing, such as establishing and disconnecting logical links. NETACP is a task that may or may not be currently in memory.
3. XPT calls the DLC layer (EPM or DDCMP).
4. XPT calls DCP. DCP implements the DDCMP protocol to ensure an error-free path, adds its header to the block of data, and sends the data to CEX.
5. CEX calls the appropriate DDM for the line selected to transmit the data. The DDM then transmits the data to the physical device and to the line.

Figure 8–3: A Basic DECnet-RSX System



8.2.2 DECnet Physical Communications Devices

A DECnet-RSX system can support more than one communications device simultaneously. Figure 8–4 illustrates several representative communications devices: KDZ, DUP, DZ, DMC, PCL, and the Ethernet devices DEUNA, DEQNA, DELUA, and DELQA. The name of the DDM is the same as that of the physical device that the DDM drives, except for the KDZ (the physical device KMC and DZ) and the DMC (the physical device DMR). Although it is not typical to have all of these devices connected to one node, the node can handle them all satisfactorily.

- **KDZ, DUP, and DZ.** These DDMs interface with DCP. This arrangement permits each DDM to share the code that maintains the DDCMP protocol.
- **DMC and PCL.** These DDMs interface directly to ECL because they do not need DCP support. DMC drives the DMR device that supports a hardware version of DDCMP. PCL drives the PCL device, which uses a non-DDCMP hardware link protocol to establish its connection, and allows a number of nodes to be connected in a local network.
- **DEUNA, DEQNA, DELUA, and DELQA.** These DDMs interface to EPM, the Ethernet Protocol Manager.

Both the DMC and PCL DDMs perform functions normally associated with DLCs and DDMs.

8.2.3 The Routing Control Processor

The routing control processor (RCP) determines the best routing path for messages by maintaining and updating the routing database. RCP exists on routing nodes only. There are two tables:

- The minimum hops/minimum cost table
- The reachability and output adjacency (ROA) table

RCP maintains and updates both tables at the same time whether messages are received by non-Ethernet or Ethernet devices using the channel.

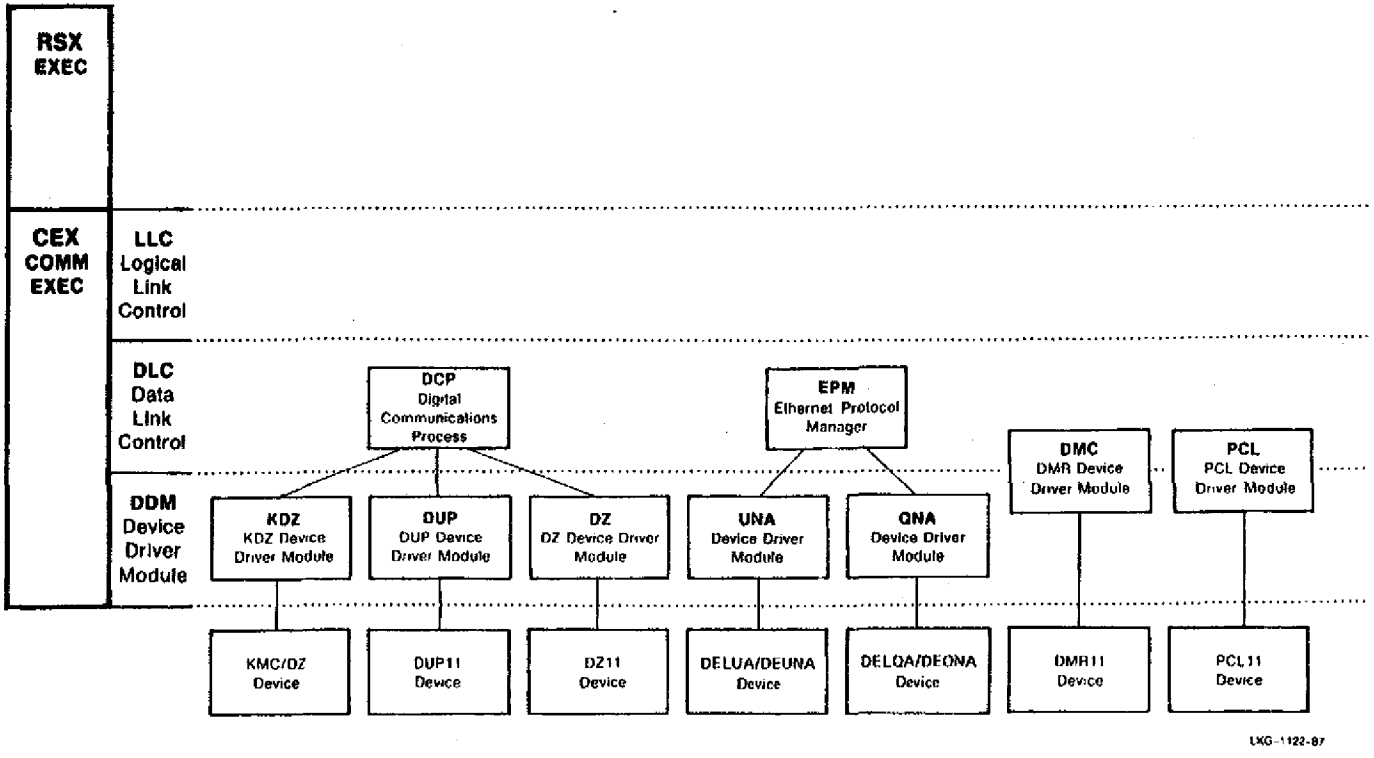


Figure 8–4: Physical Communications Devices for a DECnet-RSX System

8.2.3.1 Minimum Hops/Minimum Cost Table

Each node within the minimum hops/minimum cost table has multiple entries. When RCP receives a routing message for a particular node or group of nodes, it goes to the particular entries for each node on the channel that has received the routing message and fills in the appropriate information. For example, if RCP receives a routing message that affects one node, RCP updates one entry. If RCP receives a routing message that affects 50 nodes, RCP updates 50 entries.

To calculate the correct routing path, it scans across all entries in the channel one row at a time to find the minimum hops/minimum cost value. When the minimum hops/minimum cost value is greater than the value received in the routing message, RCP recalculates that row and fills in the new value in the appropriate entry. If the opposite is true, RCP will not change anything. RCP ensures that the minimum hops/minimum cost value is never greater than the value that was assigned to your system.

8.2.3.2 Reachability and Output Adjacency (ROA) Table

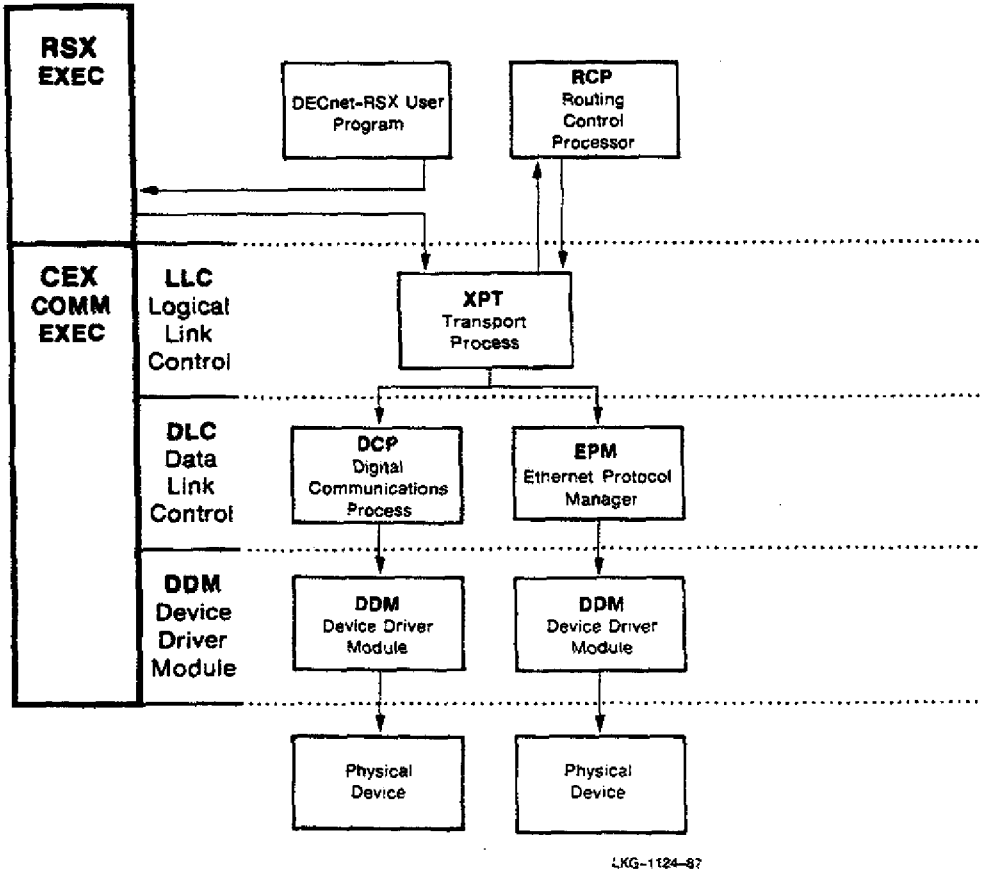
At the same time RCP calculates the minimum hops/minimum cost value, it also saves this value by recording it in the ROA table. It is the function of the ROA table to maintain the minimum hops/minimum cost value for all entries and every node using the channel. For each channel, there is a corresponding adjacency database entry. All adjacency database addresses are stored in the ROA table.

Whenever DECnet nodes discover any change that would affect network routing, a routing message is sent. For example, if a circuit goes down, a node will send out routing messages indicating that fact. The sequence of processing a routing message is:

1. A node sends a routing message.
2. When DECnet-RSX receives the routing message, it passes the message to XPT.
3. XPT calls RCP.
4. RCP updates the routing database.
5. When routing messages cause RCP to change the routing database, RCP sends a routing message to all adjacent, full-routing nodes. This process continues until all routing databases are accurate and up-to-date.

Figure 8–5 illustrates a DECnet-RSX system with routing, and Figure 8–6 illustrates the function of the hops/cost database.

Figure 8–5: DECnet-RSX System Illustrating Routing



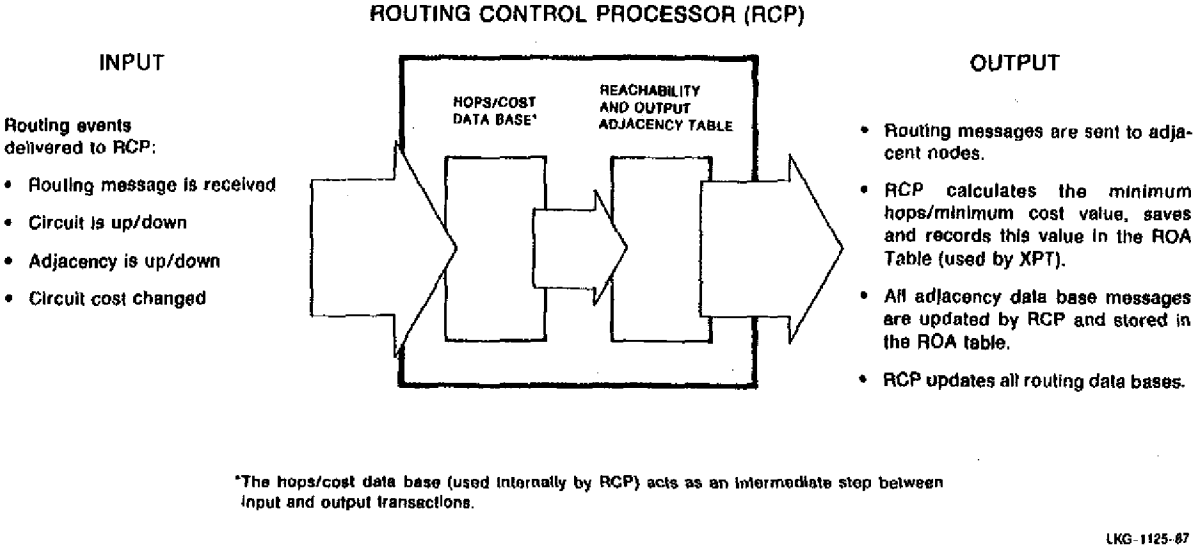


Figure 8–6: Hops/Cost Database

8.3 CEX Support Components

Several components perform support functions for CEX and are available for all CEX systems including PSI. These functions are:

- Direct line access
- Network loading
- Event logging

8.3.1 The Direct Line Access Controller

The Direct Line Access Controller (DLX) allows user-written programs that use the DLX QIO interface to transmit data over a line to another DLX program without establishing a logical link. Figure 8–7 illustrates DECnet-RSX operations with DLX.

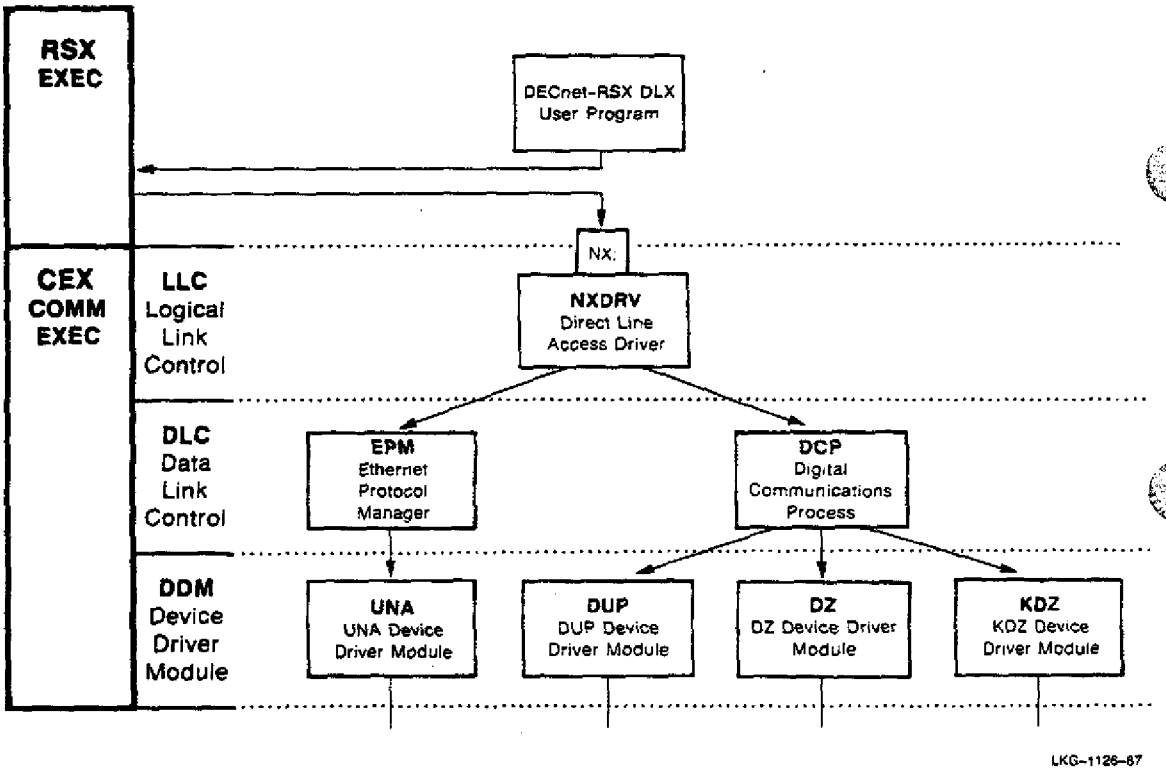
To use DLX, you issue QIO calls to the NX: device driver. QIO calls are then directed to the NX: device driver for processing.

DLX is required for RSX–11M–PLUS systems and is optional for RSX–11M/11S systems. A node requires DLX to perform the following functions:

- Down-line system loading
- Up-line system dumping
- Line/circuit loopback testing
- Console carrier requester (CCR) support

For more information on DLX QIO calls using Ethernet and non-Ethernet devices, see the *DECnet-RSX Programmer's Reference Manual.*

Figure 8–7: DECnet-RSX Operations with DLX



8.3.2 Network Loading Components

The components needed by the Network Control Program (NCP) to load the network using the NCP SET SYSTEM command include:

- The network initializer (NTINIT) task
- The network loader (NTL) task
- The Communications Executive (CEX)
- Network management volatile ancillary control processor (NMVACP).

Figure 8–8 illustrates the process of loading the network using the NCP SET SYSTEM command. The sequence of processing is:

1. NCP calls NTINIT.
2. NTINIT calls NTL. NTINIT supervises the process while NTL performs several functions. NTL loads the Communications Executive (CEX.TSK—applicable for DECnet–11M/S only) and the Communications Executive table (CETAB.TSK—created from the CETAB.MAC file).
3. NTL loads CEX.TSK (for DECnet–11M/S only) and CETAB.TSK, scans the CETAB.MAC file, and creates the network pool partition. The CETAB.TSK serves as a template. NTL scans the CETAB.MAC file and loads specific parameters into the CETAB template.
4. When NTL completes these functions, NTINIT calls NTL for each process marked for load and each line marked for load.
5. NTL loads each process marked for load, then loads lines marked for load. NTL also creates and loads the network pool, loads each process marked for load, and loads lines marked for load.

Figure 8–8: Loading the Network

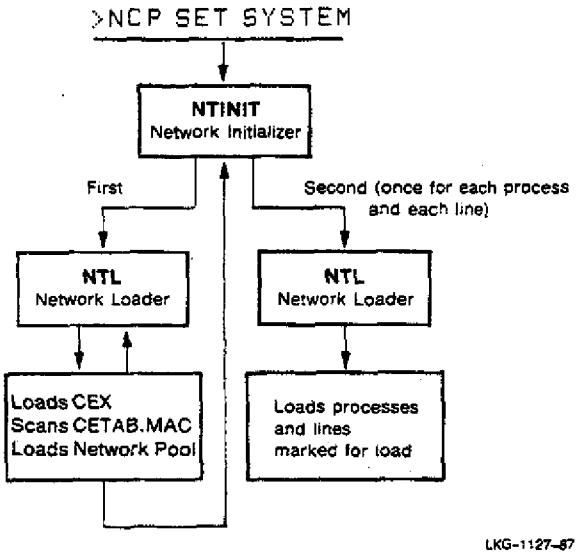


Figure 8–9 illustrates the process of turning on the network using the NCP SET EXECUTOR STATE ON command. The sequence of processing is:

1. You issue the appropriate NCP command.
2. NCP issues a QIO call to NM: (the pseudodevice).
3. NM: passes the request to NMVACP.
4. NMVACP then invokes NTINIT to start and run NETACP, the network ancillary control processor.

Figure 8–9: Turning On the Network

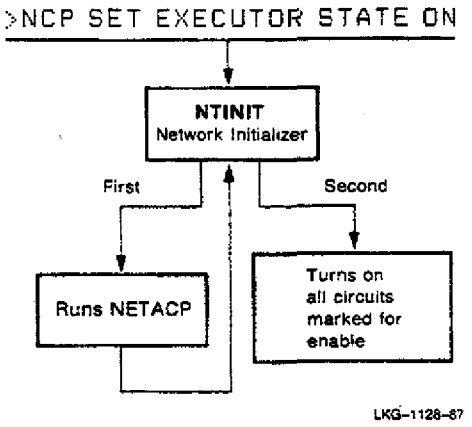
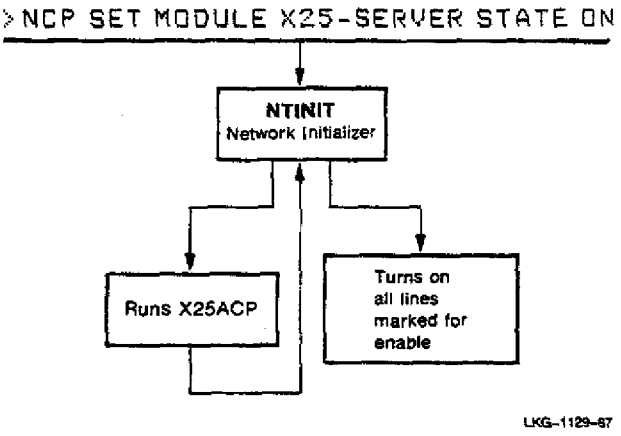


Figure 8–10 illustrates the process of turning on the X.25 Server using the NCP SET MODULE X25-SERVER STATE ON command. The sequence of processing is:

1. You issue the appropriate NCP command.
2. NCP issues a QIO call to NM: (the pseudodevice).
3. NM: passes the request to NMVACP.
4. NMVACP invokes NTINIT.
5. NTINIT starts X.25ACP running. This enables all PSI lines/circuits that are marked for enable.

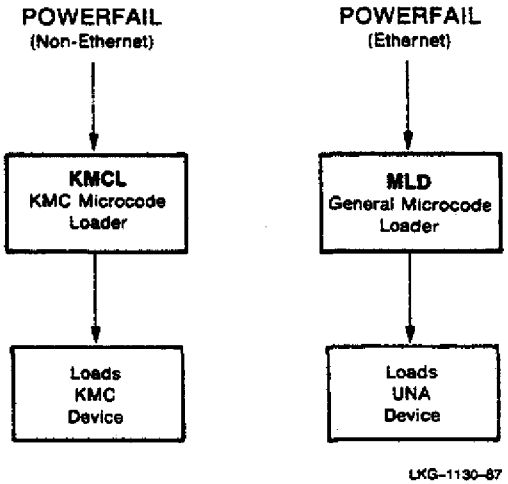
Figure 8–10: Turning On the X.25 Server



The KMC microcode loader (KMCL) loads all KMC devices (KDP, KDZ, KMX) with the proper microcode. Figure 8–11 illustrates KMCL. KMCL is invoked during powerfail recovery only on RSX–11M/M–PLUS systems. NTL loads the microcode when the line is loaded (for example, NCP SET LINE KDZ-0-0). KMCL always loads the microcode on RSX–11S systems.

The general microcode loader (MLD) loads the proper microcode for UNA devices. MLD is invoked by the device driver when a circuit is enabled.

Figure 8–11: Loading KMC Devices with Microcode



8.3.3 Event Logging Components

DECnet-RSX monitors a wide range of network activities called events. Each event represents a significant occurrence in the network, the reporting of which may prove useful in monitoring network performance. Each event is identified by a unique number, known as an event ID. For example, event 4.0 always refers to an aged packet loss. Internally, DECnet refers to events by event ID. The following is a partial list of typical network events:

0.6	Passive loopback
0.7	Aborted service request
2.1	Access control failure
3.0	Invalid message
4.0	Aged packet loss
4.1	Node unreachable packet loss
4.3	Oversized packet loss
4.6	Verification reject
4.7	Circuit down, circuit fault
4.10	Circuit up
5.0	Locally initiated state change

DECnet allows a user such as a system manager on one node to monitor events not only on local nodes, but also on remote nodes. The node at which the event takes place is known as the source node; the node to which a record of the event is transmitted is known as the sink node. See Section 2.6 for a detailed explanation of events.

Event logging components, as illustrated in Figure 8–12, include:

- The event collector (EVC)
- The Event File Interpreter utility (EVF)
- The event logger (EVL)
- The event-logging receiver (EVR)

In addition to collecting network events, EVC can apply event filters to network events. An event filter is a mechanism that allows you to select those events you want to process and to ignore others. Filters are always applied at the source node. For example, you can use an event filter to determine that an event is to be logged each time a specific circuit goes down on the system. The filter can be set so that only this event will be logged by EVC.

Each time EVC is called, it empties EVL's buffers, applies any event filters, and formats the events appropriately for the type of logging selected.

EVL logs and records network events. EVC collects the events logged by EVL and sends them to the appropriate sinks. If a remote DECnet-RSX node has EVR, the remote node can receive network events from EVC.

EVF is part of the event logging facility provided by DECnet-RSX. The facility enables you to collect events in a machine-readable file for later formatting by EVF. The event collector and Event File Interpreter are similar to the error logger and error report generator provided by RSX–11M–PLUS.

EVF is a non-privileged utility. Therefore, any user can create a formatted event listing with EVF.

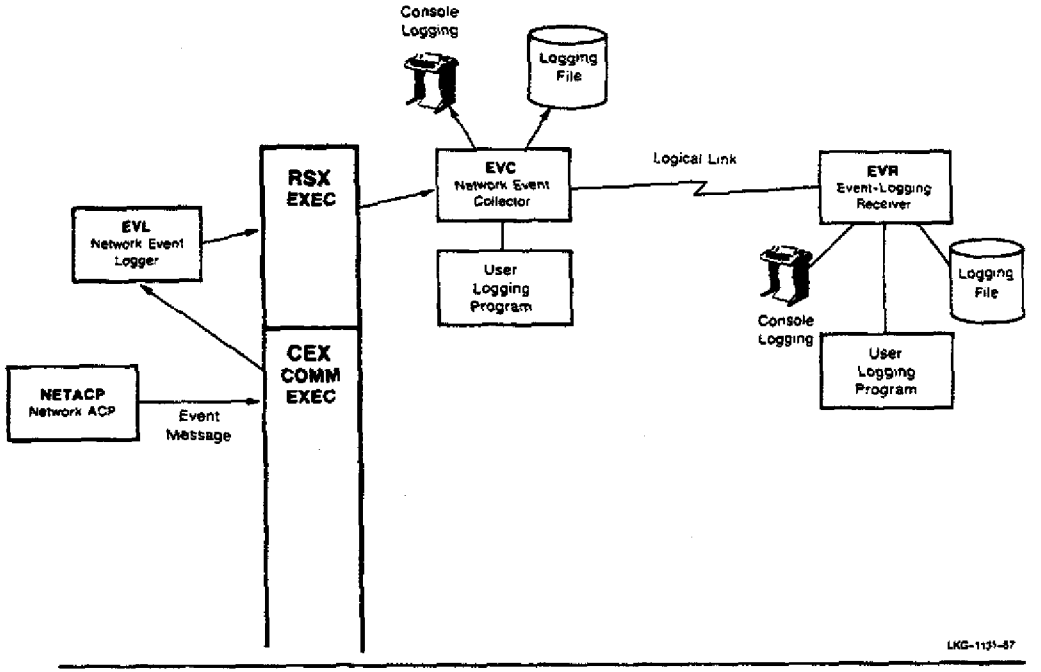
Local event logging. For local event logging, EVC can log events to a console, to a user-written logging program, or to a logging file. The logging file can be read by another user-written program.

Remote event logging. For remote event logging, EVC establishes a logical link with the remote network event logging-receiver (EVR) task. If the remote node is a DECnet-RSX node, it must have an EVR task to receive the events. EVC sends EVR records of selected events using the Network Information and Control Exchange (NICE) protocol. See the *DECnet-RSX Network Generation and Installation Guide* for a description of the NICE protocol.

The remote EVR task then has the same logging options as the local EVC task. It can log events to a console, to a user-written logging program, or to a logging file. Events can also be sent to other non-DECnet-RSX systems to be logged using object type 26. The sequence of processing a network event is:

1. NETACP sends event information to CEX.
2. CEX calls the network event logger (EVL) to handle logging the event.
3. If EVL receives one or more events, it requests the RSX Executive to call EVC to process the snapshots created by EVL.

Figure 8–12: Event Logging Components



8.4 System Management Utilities

DECnet-RSX provides three utilities for modifying network parameters and monitoring local and remote nodes on the network. These utilities are described in the following sections.

8.4.1 Modifying Network Parameters

Three utilities that modify network parameters include:

- The Network Control Program (NCP)
- The Configuration File Editor (CFE)
- The Virtual Network Processor (VNP)

8.4.1.1 The Network Control Program

NCP modifies the volatile database which consists of the various memory-resident data structures used by the system. Changes made through NCP are immediately effective on the local node. Since NCP affects only the volatile and not the permanent database, changes made through NCP are lost the next time the network is loaded.

8.4.1.2 The Configuration File Editor

CFE changes the permanent database stored in the configuration file CETAB.MAC. Changes made through CFE do not become effective until the next time the network is loaded.

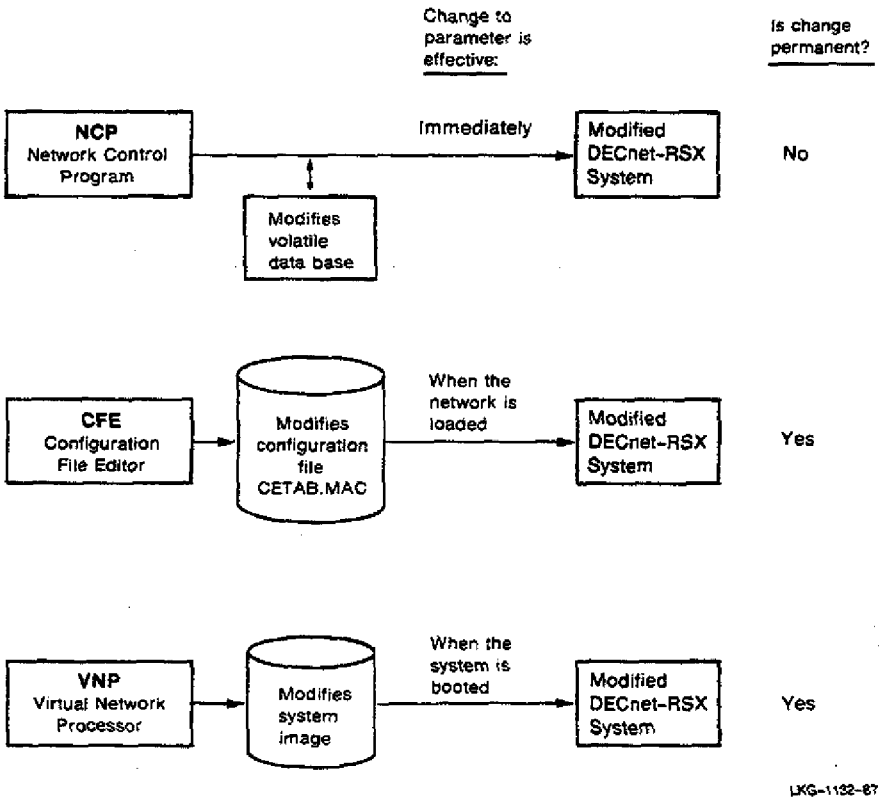
8.4.1.3 The Virtual Network Processor

VNP changes the system image of the network contained in a file called RSX11M.SYS for RSX–11M/M–PLUS systems and RSX11S.SYS for RSX–11S systems. Users on RSX–11M/M–PLUS or VMS systems must use VNP to make direct changes to the network in the system image of an RSX–11S system on disk. VNP can also be used to alter the system image file for RSX-IIM/M–PLUS systems.

Once the system image has been changed on an RSX–11S system, it can be down-line loaded or software booted. The change is permanently incorporated in the system image file and becomes effective when the system is booted.

Figure 8–13 illustrates NCP, CFE, and VNP operation. CFE modifies the CETAB.MAC file, and VNP modifies the system image file. Refer to the *DECnet-RSX Guide to Network Management Utilities* for information on using NCP, CFE, and VNP.

Figure 8–13: NCP, CFE, and VNP Operations



8.4.2 Using NCP on the Local Node

NCP on a local node communicates with two other components: the network management device driver (NMDRV) and the network management volatile ACP (NMVACP) task. NMDRV collects all information required to execute a user command, validates addresses of buffers, and checks to see that NMVACP is installed. The sequence of operations is:

1. A user on a local node issues an NCP command.
2. NCP validates the action to be performed.
3. NCP sends a QIO call to NM: (the pseudodevice) to tell NMDRV what is to be performed.
4. NMDRV sends the message to the privileged task NMVACP.
5. NMVACP actually performs the action requested by the user, such as turning on the network, turning off a circuit or changing a given network parameter. NMVACP can directly affect any portion of the volatile database.

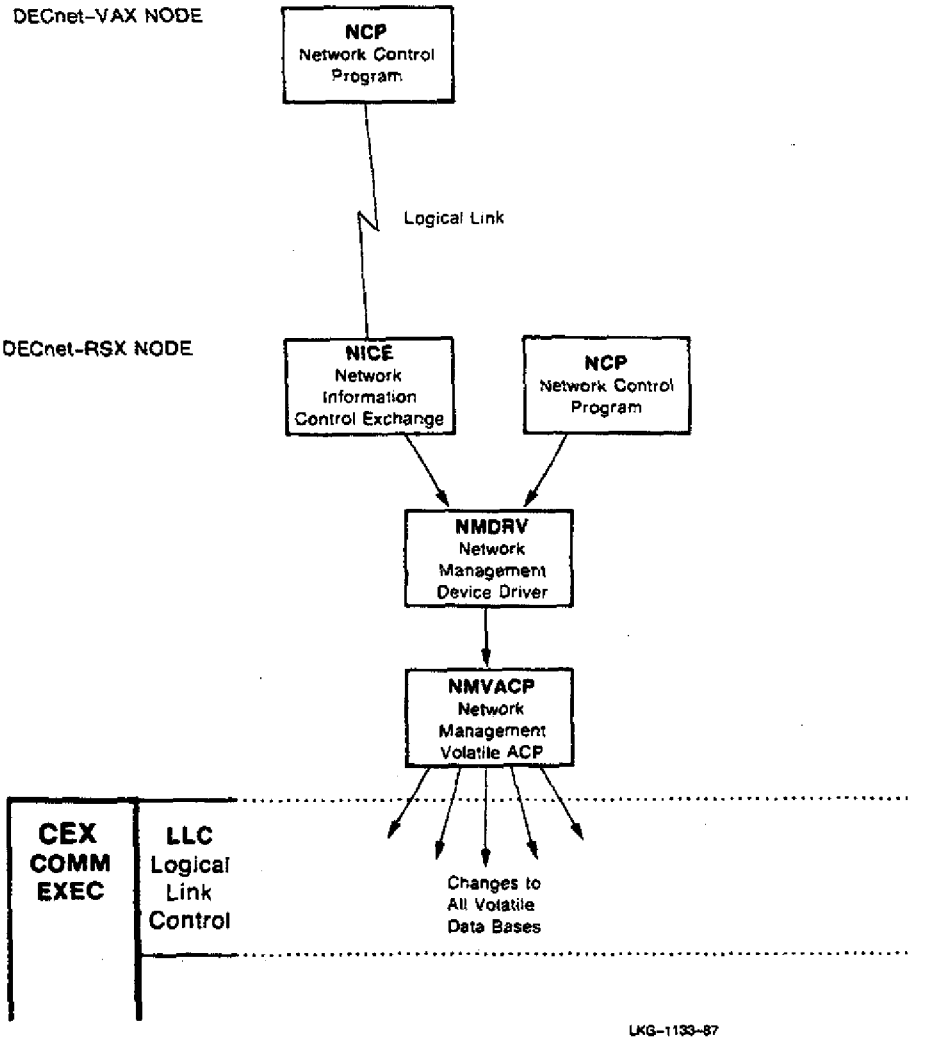
NMVACP can respond to a number of NCP requests from different users simultaneously. It maintains all relevant parameters for each user request in a separate area of its buffer space. If it runs out of available buffer space, and if it is installed as a checkpointable task, it requests an extension of its address space from the RSX Executive.

8.4.3 Using NCP on the Remote Node

NCP on a remote node can also make changes to the volatile data base through communication with the Network Information and Control Exchange (NICE) task on the local node. Figure 8–14 illustrates NCP, NICE, and related components. The sequence of processing is:

1. A user on a remote node issues an NCP command.
2. NCP validates the action to be performed.
3. If an NCP command requests that future commands be executed on a remote node, the remote NCP establishes a logical link with the Network Information and Control Exchange (NICE) task on the specified remote node.
4. When NICE receives the request from the remote NCP, it then passes the request on to NMDRV. NMDRV then passes the request to NMVACP to perform the action requested, as previously described.

Figure 8–14: NCP, NICE, and Related Components



8.5 System Management Support Components

A number of DECnet components assist in managing a given node and the network to perform such functions as network display, network verification, crash dump analysis, and loopback testing.

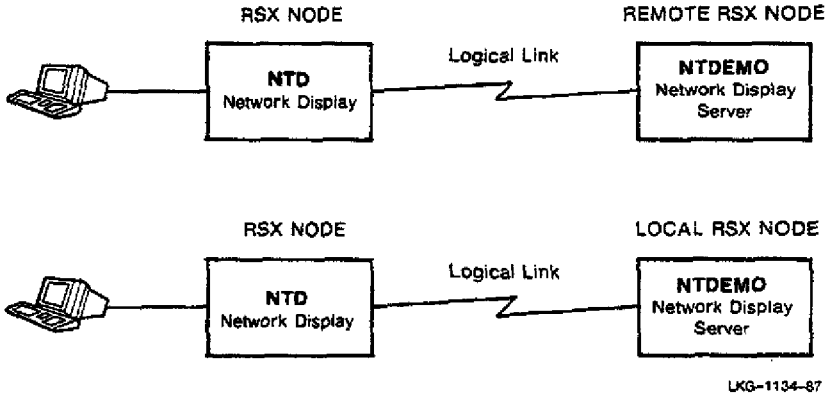
8.5.1 Network Display Utility

The Network Display (NTD) utility allows the user to monitor the current state of a node. NTD can display a node's current status for: circuits, network tasks, data structure usage, and remote node reachability. NTD uses the Network Display Server task (NTDEMO) to obtain information from a local or remote node. NTD requests and displays the information on the local terminal. Figure 8–15 illustrates NTD. The sequence of processing is:

1. The user issues an NTD command and specifies the name for the node he wants to monitor. If the node name is not given, it will default to the local system.
2. NTD establishes a logical link with NTDEMO on the requested node.
3. NTDEMO obtains the values for the appropriate information and transmits them back to NTD.
4. NTD formats the information and displays it on the local terminal.

For more information on NTD, see the *DECnet-RSX Guide to Network Management Utilities*.

Figure 8–15: Network Display Utility



8.5.2 Network Verification Program

The Network Verification Program (NVP) protects a node from unauthorized access. Network verification is available on RSX–11M/11M–PLUS nodes with multiuser protection. NVP uses the RSX system account file to verify the incoming connect's access control information and provides the target network tasks with the account's UIC, default device, and directory information. There are two levels of verification: node level verification and object level verification.

Node level verification. Node level verification has two states: ON and OFF. If OFF, no verification takes place. If ON, verification depends on the object type.

Object level verification. Object level verification has three possible levels for each object type: ON, OFF, or INSPECT. Logical links established through task names are always assigned object type 0.

Figure 8–16 illustrates network verification at the object level. The remote user program issues a connect request to establish a logical link with a user program on the local node. The request is passed to NETACP. NETACP checks the verification state in effect for the object type associated with the user program. The action taken for each state is summarized in the following table.

Table 8–1: Network Verification State

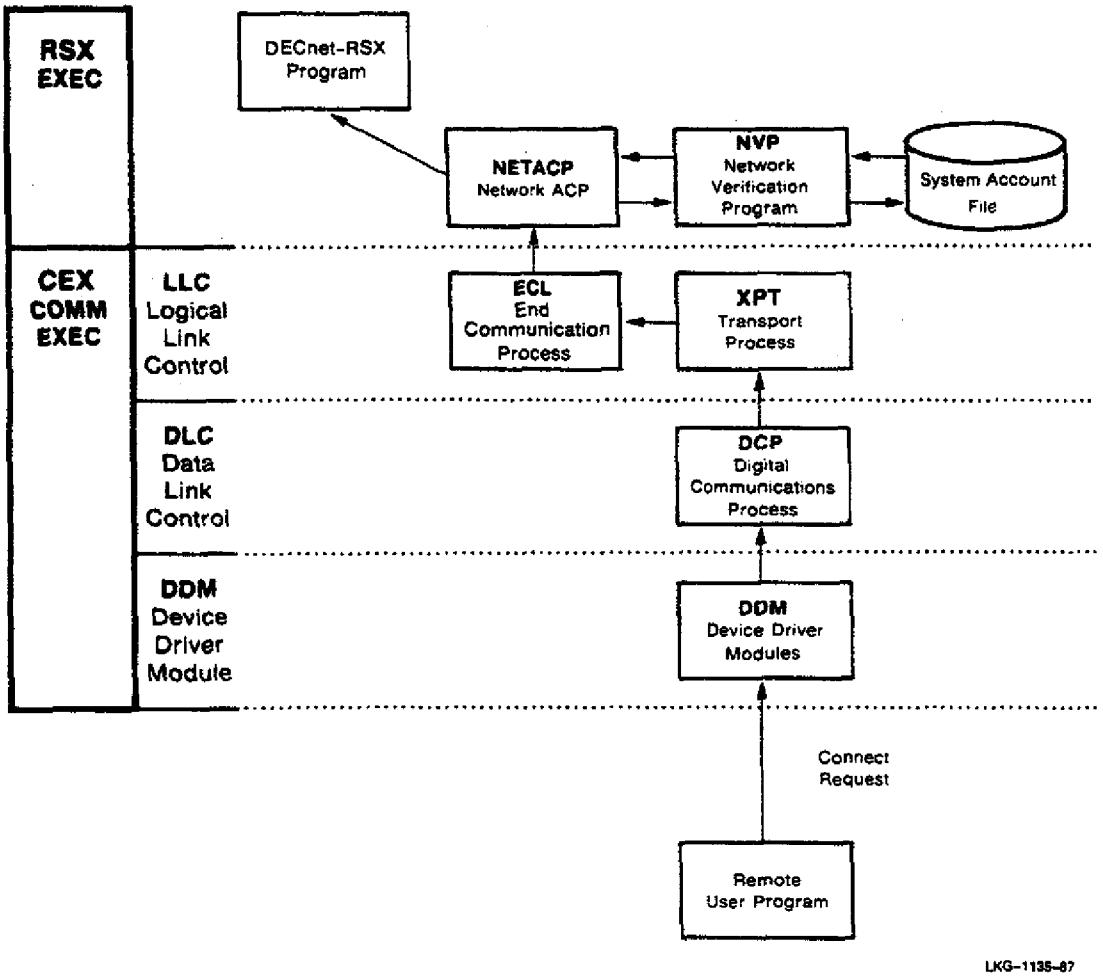
Verification	Message Passed to NVP?	Action Taken
OFF	No	None. The connection information is given to the program without any processing.
ON	Yes	Request checked and the connection rejected if information does not contain a valid user ID and password.
INSPECT	Yes	Request checked and the results passed on to the user program.

If INCOMING PROXY is enabled, the NVP can verify the connect by using the proxy information that the remote system supplied.

If the user ID and password are valid, NVP erases the password to protect confidentiality, and returns the request to NETACP. The request is then sent to the user program.

If verification is set to INSPECT and the user ID and password are invalid, the receiving user program is informed that the verification inspection failed. It is then up to the user program to take the necessary action.

Figure 8–16: Network Verification Program



8.5.3 Network Crash Dump Analyzer

The Network Crash Dump Analyzer (NDA) provides the system manager with a tool that can analyze a system crash dump and print the network databases in user-readable format in much the same way the RSX Crash Dump Analyzer (CDA) does for the system databases. Figure 8–17 illustrates NDA.

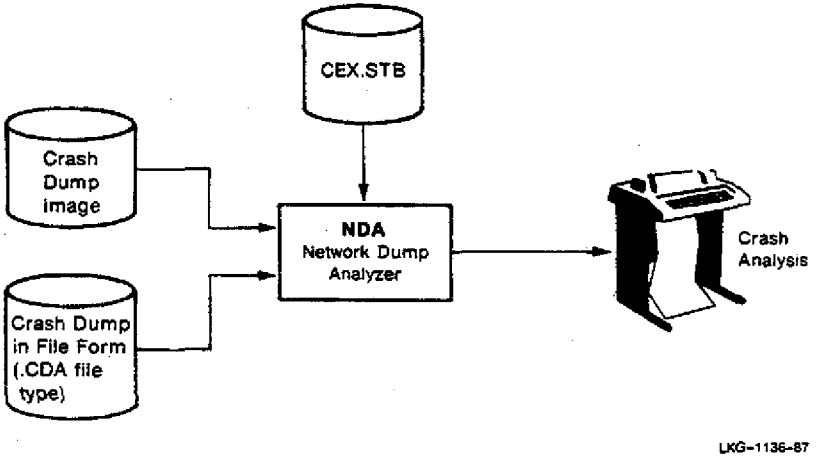
You have the option of selecting CDA during system generation (SYSGEN). If you select this option, the system will write the contents of memory to the crash dump device in the event of a system crash.

To analyze the crash dump, you must use both NDA and CDA. When you use CDA to analyze the crash dump, you convert the crash dump image into file form: *file-name*.CDA. You then produce an analysis from that file using the operating system symbol table. When you use NDA to analyze the crash dump, you analyze the crash dump image using the Communications Executive (CEX) symbol table.

The output of NDA and CDA is intended primarily for Digital software specialists. If you need the assistance of a Digital software specialist, you should submit a copy of the dump in file form, together with the symbol tables, on a medium such as magnetic tape. This allows the software specialist to run NDA, CDA and other checks on the data.

For more information on NDA, see the *DECnet-RSX Guide to Network Management Utilities* .

Figure 8–17: Network Crash Dump Analyzer



8.5.4 Loopback Testing

Loopback testing allows the user to send a message and have it returned, thus verifying that it was transmitted without errors. Variations of loopback testing include:

- Hardware loopback circuit testing
- Circuit/line level loopback testing
- Node level loopback testing

8.5.4.1 Hardware Loopback Circuit Testing

Figure 8–18 illustrates hardware loopback circuit testing for Ethernet and non-Ethernet devices. The sequence of operation for devices using DDCMP is:

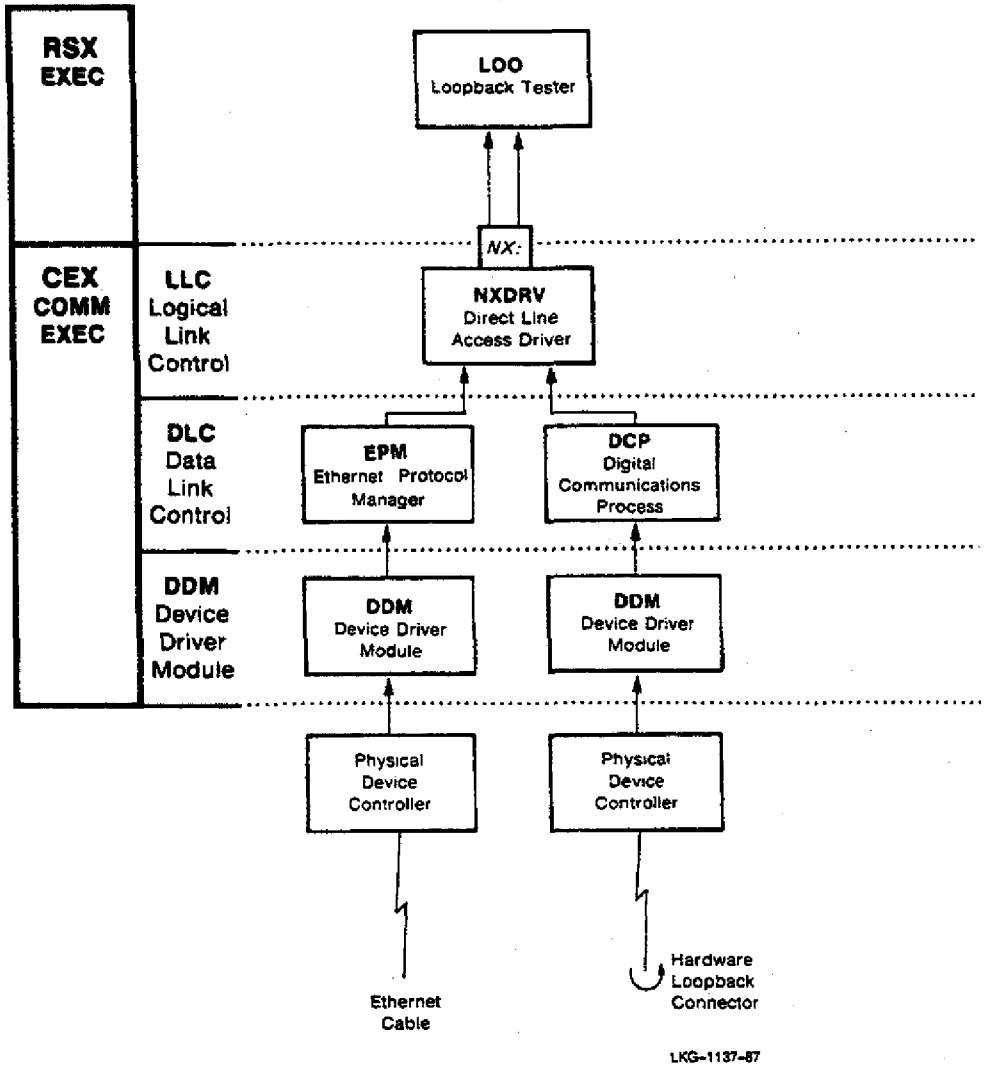
1. First place a hardware loopback connector on the line. This loopback connector can be placed at the local side or at the remote side of the line and at appropriate points along the data path in order to isolate any fault.
2. You issue the appropriate NCP command to start the loopback test.
3. NCP runs the loopback tester (LOOPER) task and LOOPER assigns a circuit to NX: (the pseudodevice).
4. LOOPER transmits test messages defined by the Maintenance Operation Protocol (MOP), tests to see if they are echoed accurately, and then displays any errors.

For devices using the Ethernet cable, no loopback connector is needed. The sequence of operation for devices using the Ethernet cable is:

1. You issue the appropriate NCP command to start the loopback test.
2. NCP assigns a circuit to NX: (the pseudodevice) and runs the loopback tester (LOOPER) task.
3. LOOPER transmits test messages defined by the Maintenance Operation Protocol (MOP), tests to see if they are echoed accurately, and then displays any errors.

For more information on hardware connectors and loopback testing, see Chapter 4 of this manual.

Figure 8–18: Hardware Loopback Circuit Testing



8.5.4.2 Circuit/Line Level Loopback Testing

Figure 8–19 illustrates circuit/line level loopback testing. This tests an entire loop between two nodes. The sequence of operation for devices using DDCMP is:

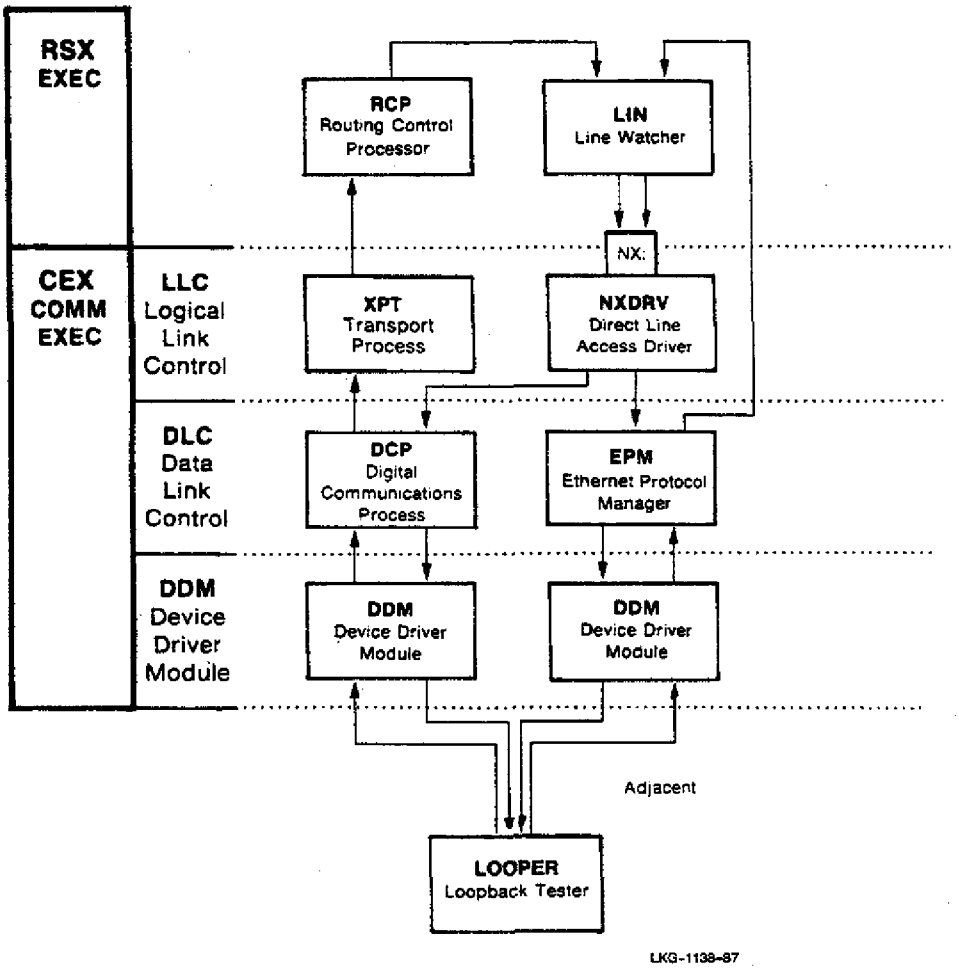
1. You issue the appropriate NCP command.
2. LOOPER issues QIO calls to NX: (the pseudodevice) to transmit test messages defined by MOP.
3. At the adjacent node, messages pass from the DDM to DCP, for software using DDCMP, to XPT.
4. XPT calls the routing control processor (RCP) task to inform other nodes that the circuit is down and to update the routing database accordingly.
5. XPT passes control to the link watcher (LIN) task. LIN reassigns the circuit to DLX and transmits any incorrectly formatted messages, thereby forcing LOOPER at the adjacent node to retransmit the test message.
6. When the message is retransmitted, LIN recognizes that this is a request for loopback testing and starts echoing the messages by way of QIOs to the NX: pseudodevice through a series of processes back to the sending node.
7. LOOPER at the sending node determines if there are any errors and reports the result of the test back to NCP, which in turn informs the user.

The sequence of operation for devices using the Ethernet cable is:

1. At the adjacent node, the message passes from the DDM to the Ethernet Protocol Manager (EPM). EPM passes control to LIN.

   Some devices (UNA) will perform the mirror function themselves. The DDM never sees the test message. The UNA handles this for loops done to a specific address. For loops to a multicast address, the message is passed by DDM to EPM to LIN. LIN performs the mirror function.
2. LIN receives the message, recognizes that this is a request for loopback testing, and starts echoing the messages by way of QIOs to NX: (the pseudodevice), EPM, and then back to the sending node.
3. LOOPER at the sending node determines if there are any errors and reports the result of the test back to NCP, which in turn informs the user.

Figure 8–19: Circuit/Line Level Loopback Circuit Testing



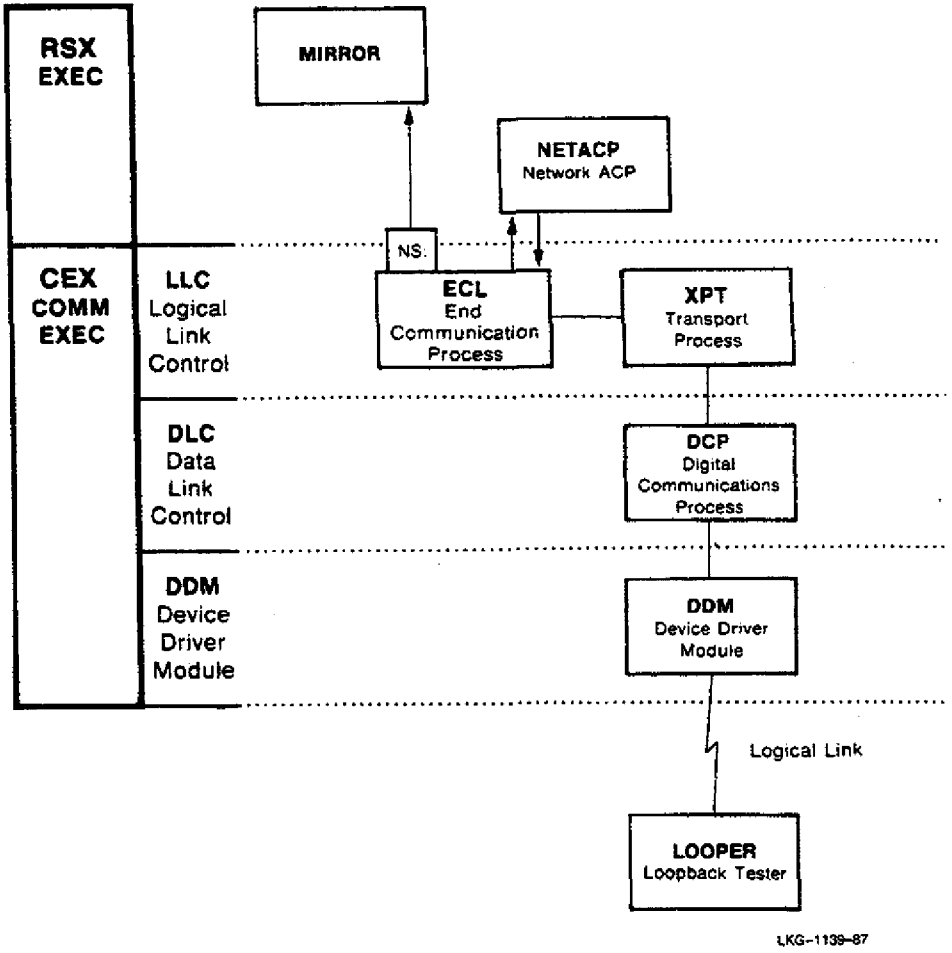
8.5.4.3 Node Level Loopback Testing

Figure 8–20 illustrates node level loopback testing which checks the ability of the network to support a logical link between any two remote nodes. The sequence of operation is:

1. You issue the appropriate NCP command.
2. NCP calls LOOPER to establish a logical link with the loopback mirror (MIR) task on the remote node using the task-to-task programming interface.
3. LOOPER sends a series of test messages. Each message passes through the DDM to DCP to XPT. XPT passes control to ECL.
4. ECL calls NETACP to perform its function and passes control back to ECL. ECL then transmits the message to the remote node.
5. After testing the logical link for the specified number of messages, LOOPER reports back to NCP, and NCP in turn informs the user of the success or failure of the test.

For information on various loopback tests, see Chapter 4 of this manual.

Figure 8–20: Node Level Loopback Testing



8.6 DECnet Utilities

DECnet-RSX provides file access services and file utilities for terminal and remote file access operations.

8.6.1 File Access Services

File access is provided by:

- The File Access Listener (FAL)
- The command file submission task (MCM)

8.6.1.1 The File Access Listener

FAL services network requests for file access and performs the following operations:

- File creation
- File retrieval
- File deletion
- File execution/submission
- File printing
- File renaming
- File directory lists

There are two versions of FAL: File Control System (FCS) FAL and Record Management System (RMS) FAL. Either FAL will perform file transfers in conjunction with NFT. RMS FAL allows RMS programs on a remote node to perform record access to sequential, relative, and indexed files on the local node. FCS FAL supports sequential file operations. You specify FAL support during network generation (NETGEN).

8.6.1.2 The Command File/Batch File Submission Task

MCM is a service task that executes or submits command files or batch jobs for NFT and FAL. MCM has two versions of operation available to users. You must choose between these two versions during network generation (NETGEN).

- One version, available for RSX–11M and RSX–11M–PLUS, allows a user to submit a file to the indirect command file processor (AT.) for execution.
- The other version, available for RSX–11M–PLUS only, allows a user to submit a file to the RSX Queue Manager for execution as a batch job.

Figure 8–21 illustrates the two versions of MCM operation. Version A illustrates file submission to the indirect command file processor. The sequence of operation is:

1. A user issues an NFT command either on a local node or a remote node to initiate the request. If issued from a local node, NFT transmits the request directly to MCM. If issued from a remote node, NFT establishes a logical link with FAL on the node that will execute the indirect command file, shown here as the local node.
2. FAL passes the command to MCM.
3. MCM submits the file to the indirect command file processor (AT.) for execution. The console terminal must be logged on for the command file to work because the indirect command file executes on a user terminal (designated as CO:).

Version B is identical up to the MCM utility. At this point, MCM submits the file to the RSX Queue Manager (QMG) for execution as a batch job, rather than to the indirect command file processor.

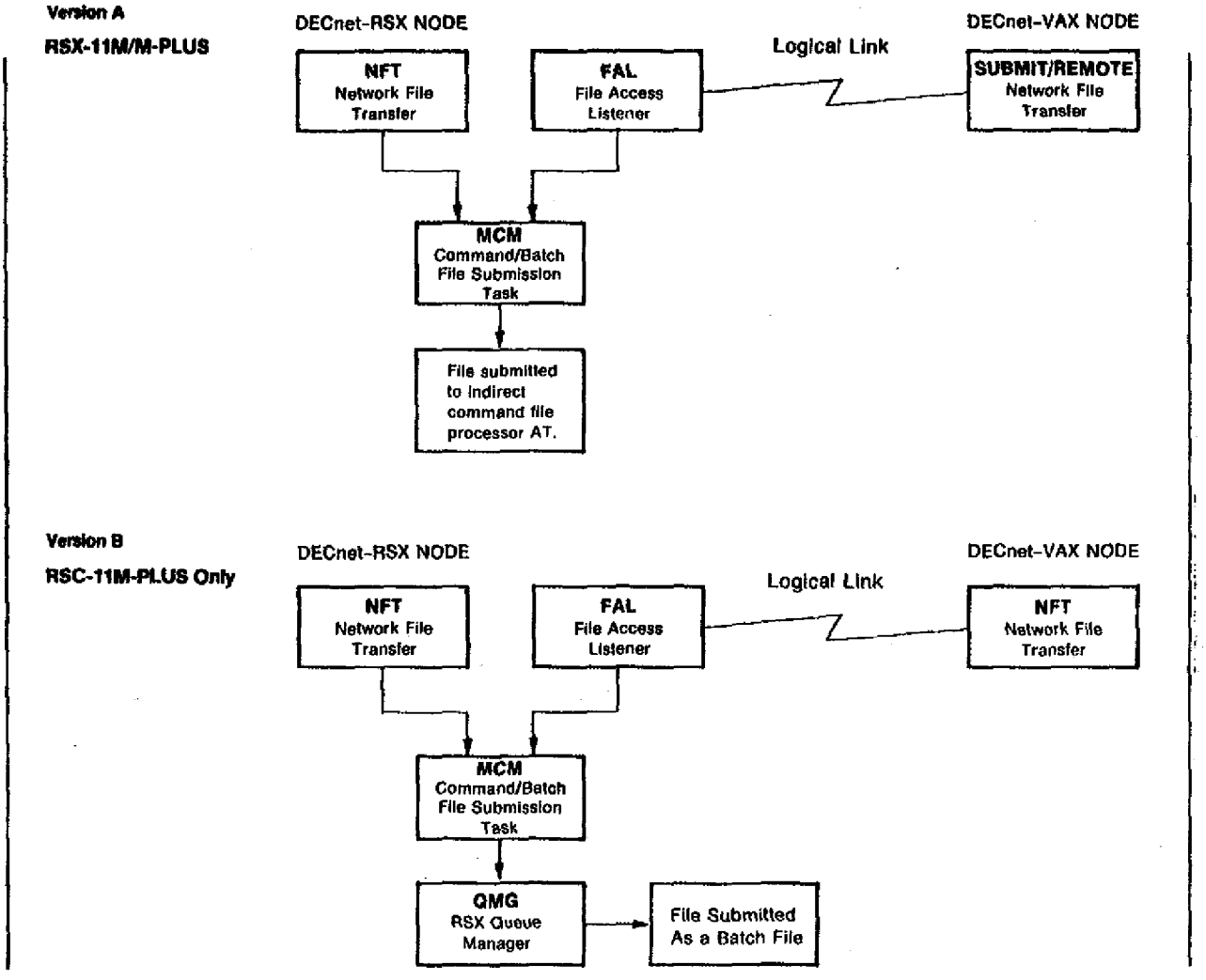


Figure 8–21: Command/Batch File Submission Task

8.6.2 File Utilities

Two utilities that allow users to access files on other nodes in the network include:

- The Network File Transfer (NFT)
- The File Transfer Spooler (FTS)

8.6.2.1 The Network File Transfer Utility

NFT communicates with FAL on the remote node and performs the following operations:

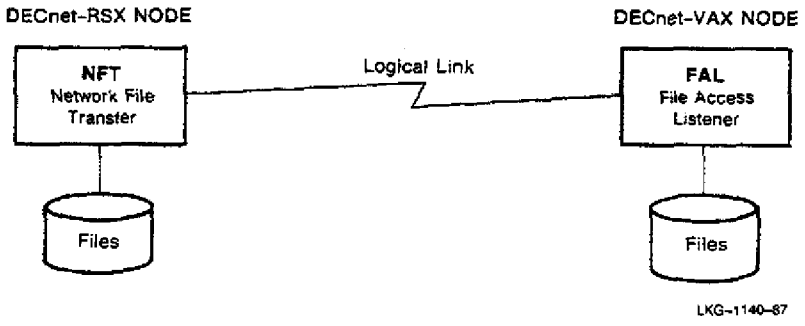
- File transfer
- File deletion
- Directory listing
- Protection changes
- Printing
- Renaming
- Command file/batch file submission

Figure 8–22 illustrates NFT. The sequence of processing is:

1. You issue an NFT command.
2. NFT determines what action is to be taken. If you are transferring a file from a remote node to a local node, NFT establishes a logical link with FAL on the remote node.
3. FAL on the remote node locates the appropriate file and performs the requested operation. In the case of a file transfer, the file's data is sent to NFT.

For more information on NFT and FAL, see the *DECnet-RSX Guide to User Utilities* and the *DECnet-RSX Network Generation and Installation Guide.*

Figure 8–22: Network File Transfer



8.6.2.2 The File Transfer Spooler Utility

FTS is similar to NFT except that it allows a series of file transfer requests to be queued and performed in sequence. Operations may execute immediately or may be queued until either the local or remote node becomes available or until a user-specified time. Figure 8–23 illustrates FTS.

FTS works in cooperation with the File Transfer Spooler Dequeuer (FTSDEQ). FTSDEQ removes requests one by one, creates a logical link on the appropriate remote node, and executes the file transfer request. If you specify FTS during network generation, FTSDEQ is automatically included.

For more information on FTS, see the *DECnet-RSX Guide to User Utilities*.

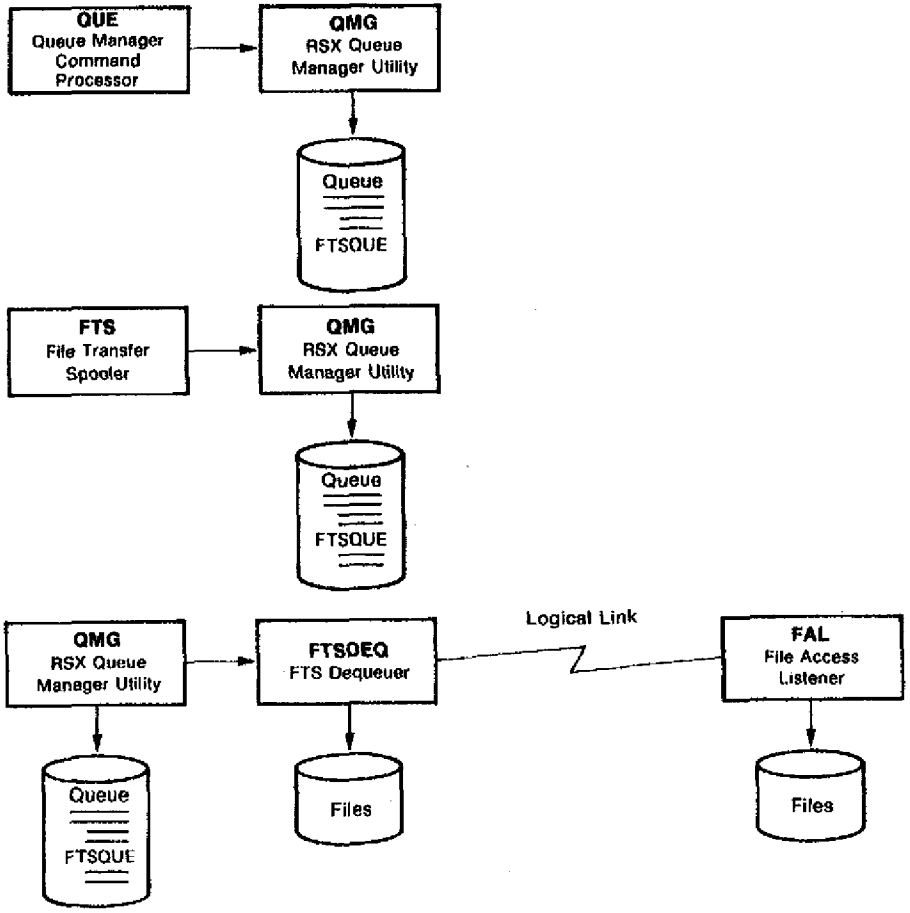


Figure 8–23: File Transfer Spooler

8.6.3 Terminal and Control Utilities

DECnet-RSX provides three utilities that allow you to log on to a remote node, send messages to a user on another node, and control a remote task. These utilities are:

- The Remote Terminal (RMT) utility
- The Terminal Communications (TLK) utility
- The Remote Task Control (TCL) utility

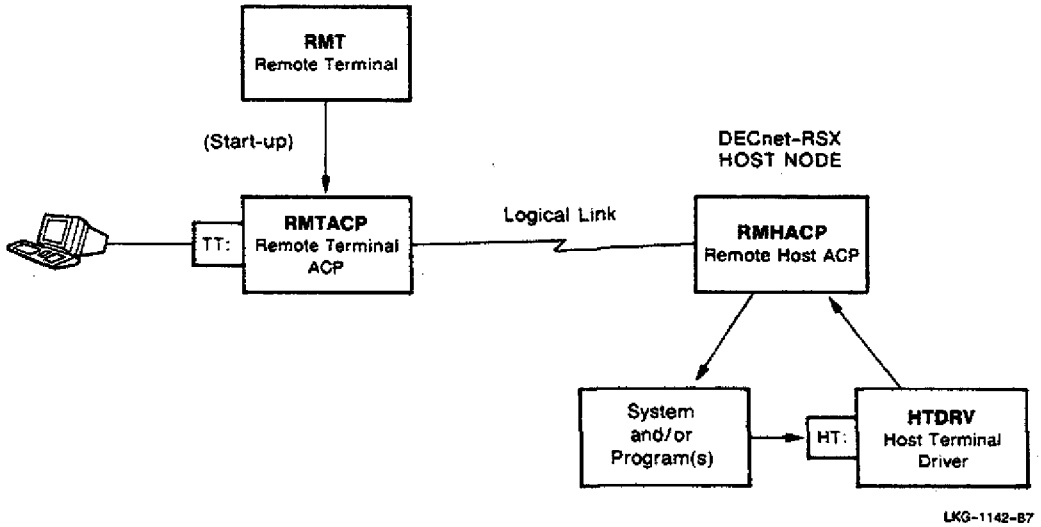
8.6.3.1 The Remote Terminal Utility

RMT allows you to log on to a remote node. Once logged on, you can execute commands as if you were actually logged on to a terminal at the remote system. RMT is available only between DECnet-RSX nodes. Figure 8–24 illustrates RMT. The sequence of operation is:

1. Log on to your DECnet-RSX local node and issue an RMT command, specifying the host node name.
2. RMT passes control to the Remote Terminal ACP task (RMTACP).
3. RMTACP establishes a logical link with Remote Host ACP (RMHACP) on a DECnet-RSX remote node. RMTACP on the local node controls your terminal and passes all data to RMHACP on the remote node.
4. RMHACP communicates with the DECnet-RSX system. To the remote system, RMHACP appears as a user typing at a terminal. Each user logged on to a given host system is assigned one unit (for example, HT0: or HT1: or HT2:), associated with the Host Terminal Driver (HTDRV). Thus, while RMHACP communicates directly with the system, the system must go through HTDRV when communicating with RMHACP.

The result of this sequence is that your terminal on the local node acts as if it were directly connected to the remote system. All commands are executed on the remote system. When you log off, the logical link is disconnected and control returns to your original system. For more information on RMT, see the *DECnet-RSX Guide to User Utilities*.

Figure 8–24: Remote Terminal Utility



8.6.3.2 The Terminal Communications Utility

TLK allows users to engage in an interactive dialog or to send single-line messages to another user on a remote DECnet-RSX node. The remote node must have installed the TLK Server Task, LSN.

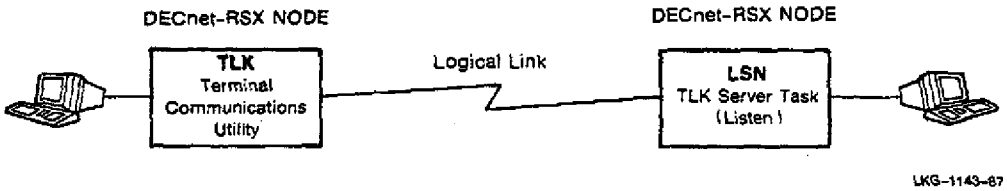
LSN allows a remote user running TLK to communicate with a terminal user on a local node. Figure 8–25 illustrates TLK.

The sequence of operation is:

1. You issue a TLK command on a local node.
2. TLK establishes a logical link with LSN on the remote node. LSN communicates with the remote user. Once the logical link is established, the user can send information back and forth.

For more information on TLK, see the *DECnet-RSX Guide to User Utilities.*

Figure 8–25: Terminal Communications Utility



8.6.3.3 The Remote Task Control Utility

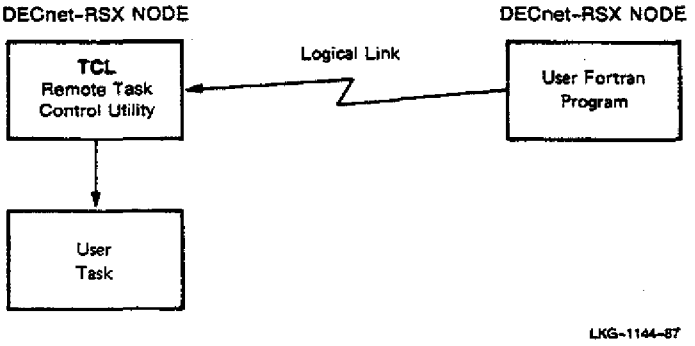
TCL allows a FORTRAN program on a remote node to control task execution on the node where TCL is installed. TCL accepts requests to run a task immediately or at some specified time, abort the current running of a task, or cancel requests for future execution of a task. Figure 8–26 illustrates TCL.

The sequence of processing is:

1. The FORTRAN program on the remote node establishes a logical link with TCL on the local node and passes the request to TCL.
2. TCL runs the specified task.

For more information on TCL, see the *DECnet-RSX Guide to User Utilities.*

Figure 8–26: Remote Task Control



8.7 Satellite Support Components

DECnet-RSX provides several support functions for DECnet–11S and server systems. DECnet–11S systems are memory-only systems that run under the RSX–11S operating system. Such systems have a wide range of applications, from a simple dedicated process control application to a more sophisticated system. A sophisticated DECnet–11S system may be similar to DECnet–11M /M–PLUS systems except that it lacks disk storage and must rely heavily on a host DECnet–11M/M–PLUS system for support.

DECnet-RSX provides the following support for DECnet–11S systems:

- Down-line system loading
- Up-line system dumping
- Down-line task loading

8.7.1 Down-Line System Loading

The down-line system loader (DLL) resides on a host DECnet–11M/M–PLUS system and is used to down-line load RSX–11S systems. The host system must be directly connected by a physical link to the DECnet–11S target system and must be able to access the target's system image file.

Figure 8–27 illustrates down-line system loading. To initiate the process, a user on the target system activates the boot procedure. The bootstrap loader in ROM sends a load request message to the DECnet–11M/M–PLUS node that will load the DECnet–11S system. The sequence of processing for devices using DDCMP is:

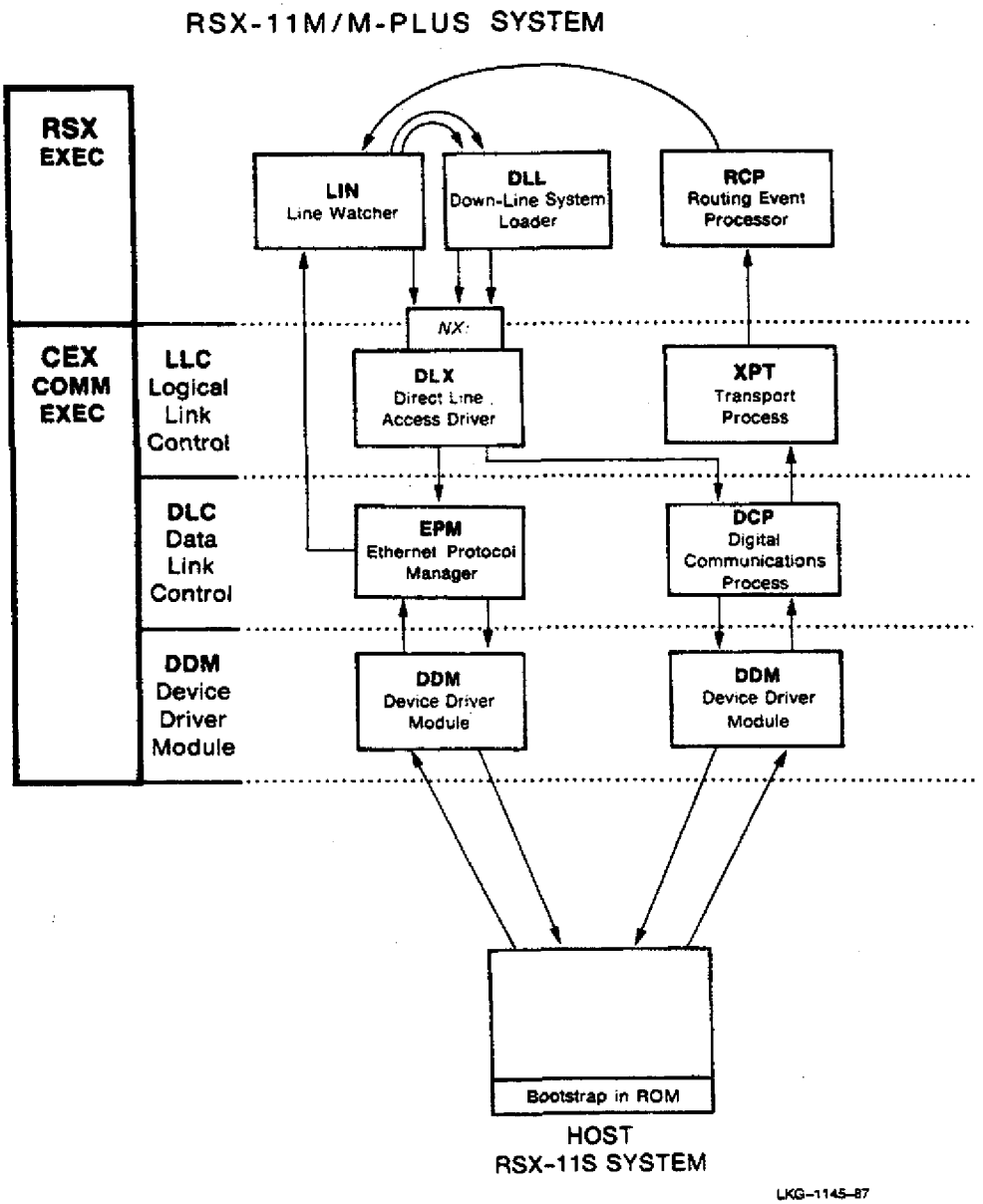
1. The load request message passes through the DDM to DCP to XPT. XPT calls RCP. RCP updates its routing database and XPT calls the line watcher (LIN) to monitor the circuit.
2. LIN changes circuit ownership to DLX, transmits load messages, receives load requests and invokes DLL.
3. DLL uses the Maintenance Operation Protocol (MOP) to perform the down-line load.

The sequence of processing for devices using the Ethernet cable is:

1. The load request message passes through the DDM to the Ethernet Protocol Manager (EPM) to LIN.
2. LIN receives a copy of the message from EPM, recognizes the message as a load request, then invokes DLL to perform the down-line load.

Before you can perform down-line system loading, you must create a database with the necessary parameters. Refer to Chapter 5 of this manual for information on how to do this.

Figure 8–27: Down-Line System Loading



8.7.2 Up-Line System Dumping

DECnet-RSX supports up-line system dumping to handle a system crash on a remote RSX-11S system. The RSX-11S memory image is transmitted up the line to an adjacent DECnet-11M/M-PLUS system to be analyzed.

To perform up-line system dumping, the RSX-11S host node must be built with the network panic dump routine (NETPAN). NETPAN is an extension of the RSX-11S Executive. In the event of a system crash, NETPAN sends a dump request message to the adjacent DECnet-11M/M-PLUS system.

Figure 8-28 illustrates up-line system dumping. The sequence of operation for devices using DDCMP is:

1. The network panic dump routine (NETPAN) sends the dump request message received by the host through the device driver module (DDM) to DCP to XPT. XPT calls RCP. RCP updates its routing database and calls the line watcher (LIN) to monitor the circuit.
2. LIN changes circuit ownership to DLX, transmits dump request messages, receives dump requests, and invokes the up-line system dumper (DUM).
3. DUM and NETPAN communicate over NX: using MOP to perform the dump.

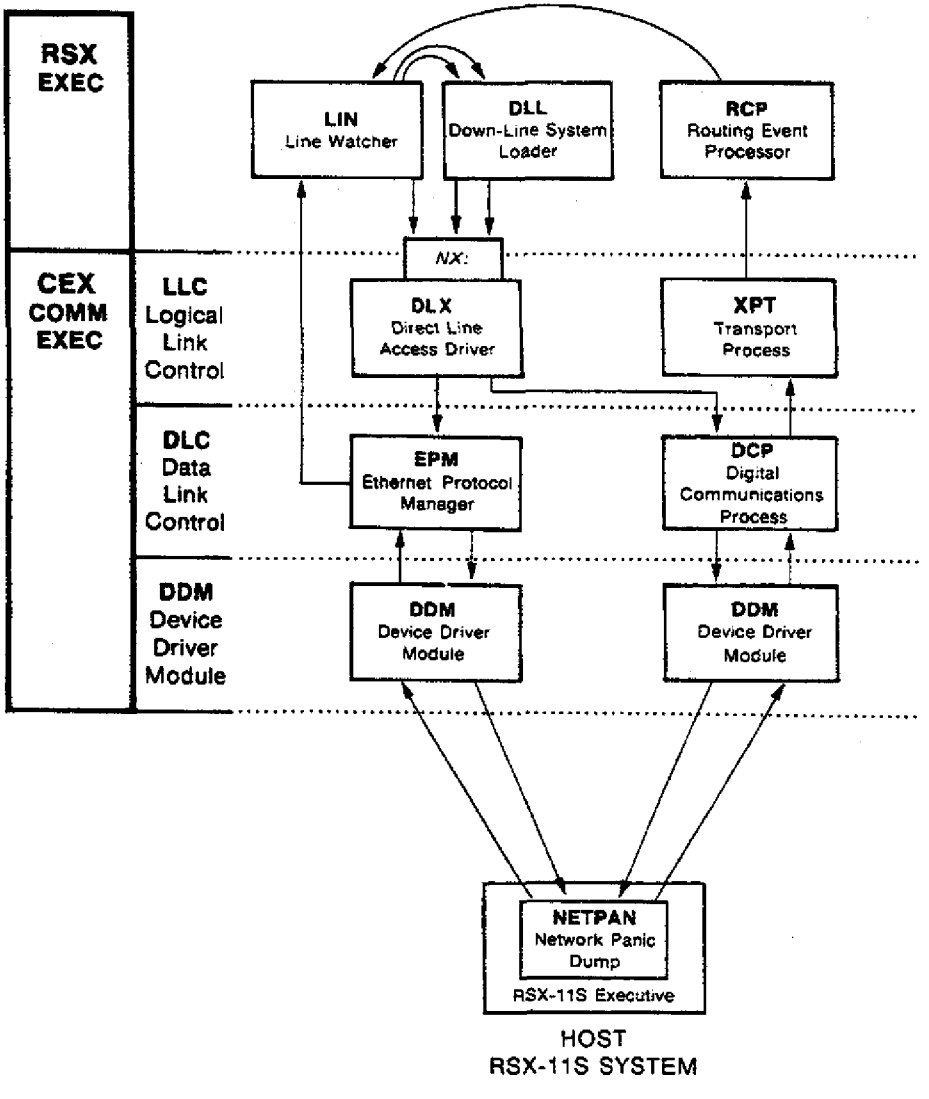
The sequence of processing for devices using the Ethernet cable is:

1. NETPAN sends the dump request message through the DDM to the Ethernet Protocol Manager (EPM).
2. LIN receives a copy of the message from EPM, recognizes the message as a dump request, then invokes DUM to perform the dump.
3. DUM and NETPAN use MOP to perform the dump.

The dump will appear in file form on the DECnet-11M/M-PLUS system. You can then analyze it using the Network Crash Dump Analyzer (NDA) or the RSX Crash Dump Analyzer (CDA).

Before you can perform up-line system dumping, you must create a database with the necessary parameters. Refer to the *DECnet-RSX Network Generation and Installation Guide* and Chapter 5 of this manual for more information on NETPAN and DUM.

Figure 8–28: Up-Line System Dumping



8.7.3 Down-Line Task Loading

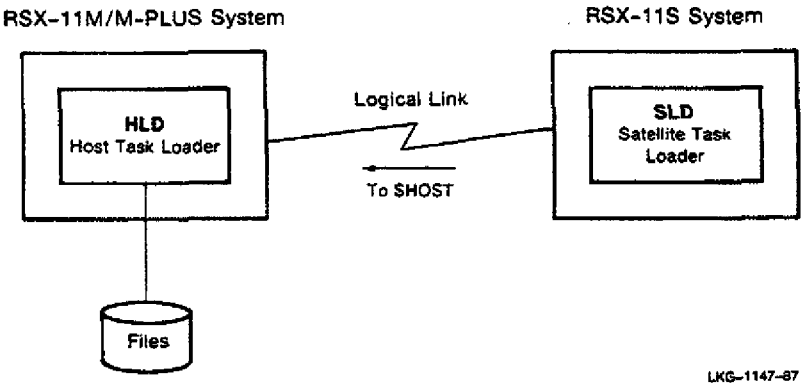
The Host Down-Line Task Loader (HLD) resides on the host DECnet–11M/M–PLUS system and down-line loads a task to a DECnet–11S node. Unlike down-line system loading, this procedure uses task-to-task communication over a logical link. HLD determines what task to load from information contained in a connect request.

Figure 8–29 illustrates down-line task loading to an RSX–11S system. Each of the two nodes is shown in a simplified diagram. Each node uses the task-to-task programming interface. The sequence of processing is:

1. A user runs a task.
2. Requests are passed to the Satellite Task Loader (SLD). SLD receives requests to load a task, to load an overlay segment, or to checkpoint a task.
3. SLD establishes a logical link with HLD, the server task on the host node.
4. When HLD receives the request to down-line load a specific task, to load an overlay segment, or to checkpoint a task, it checks its database for the location of the task image.
5. HLD then transmits the task image or the overlay segment, or it receives data to be checkpointed over the logical link to the requesting RSX–11S host node, where SLD loads the program and executes it.

Before you can perform down-line task loading, you must create a down-line task load database. Refer to the *DECnet-RSX Network Generation and Installation Guide* and Chapter 5 of this manual for more information on creating this database and using DUM.

Figure 8–29: Down-Line Task Loading



8.8 PSI Operation

The Packetnet System Interface (PSI) allows a non-DECnet PSI user program to communicate with another similar program over a Packet Switching Data Network (PSDN). PSI operation includes:

- The PSI user interface
- DECnet interface to X.25
- X.29 remote terminal support

8.8.1 The PSI User Interface

Figure 8–30 illustrates the PSI user interface. The sequence of processing for PSI is:

1. A PSI user program issues a QIO call to NW: (the pseudodevice). The RSX Executive calls a driver, X.25 user interface driver NW:, to process the data. The X.25 user interface ACP (X25ACP) assists NW: with its processing.
2. NW: communicates with an LLC such as the packet level interface (PLI). This LLC sends a message to CEX indicating it has completed its portion of the processing.
3. CEX calls a DLC, the line access procedure B (LAB). PLI communicates with LAB. LAB sends a message to CEX indicating it has completed its portion of the processing.
4. PSI calls a PSI DDM (either an SDP or SDV). Each of these 3 processes (PLI, LAB, and PSI DDM) handles one level of the X.25 protocol.
5. The PSI DDM connects to a physical device and line connected to a PSDN. While the data appears to flow in one direction, it actually passes in both directions concurrently.

Figure 8–30: PSI User Interface

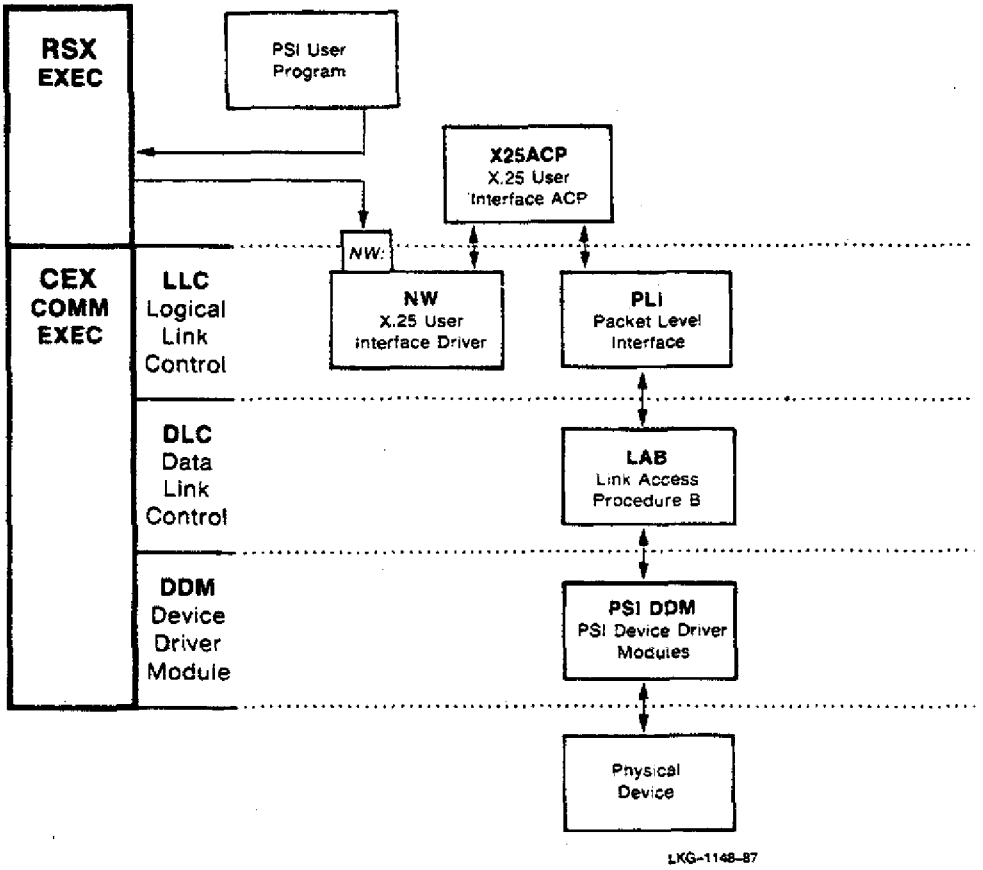
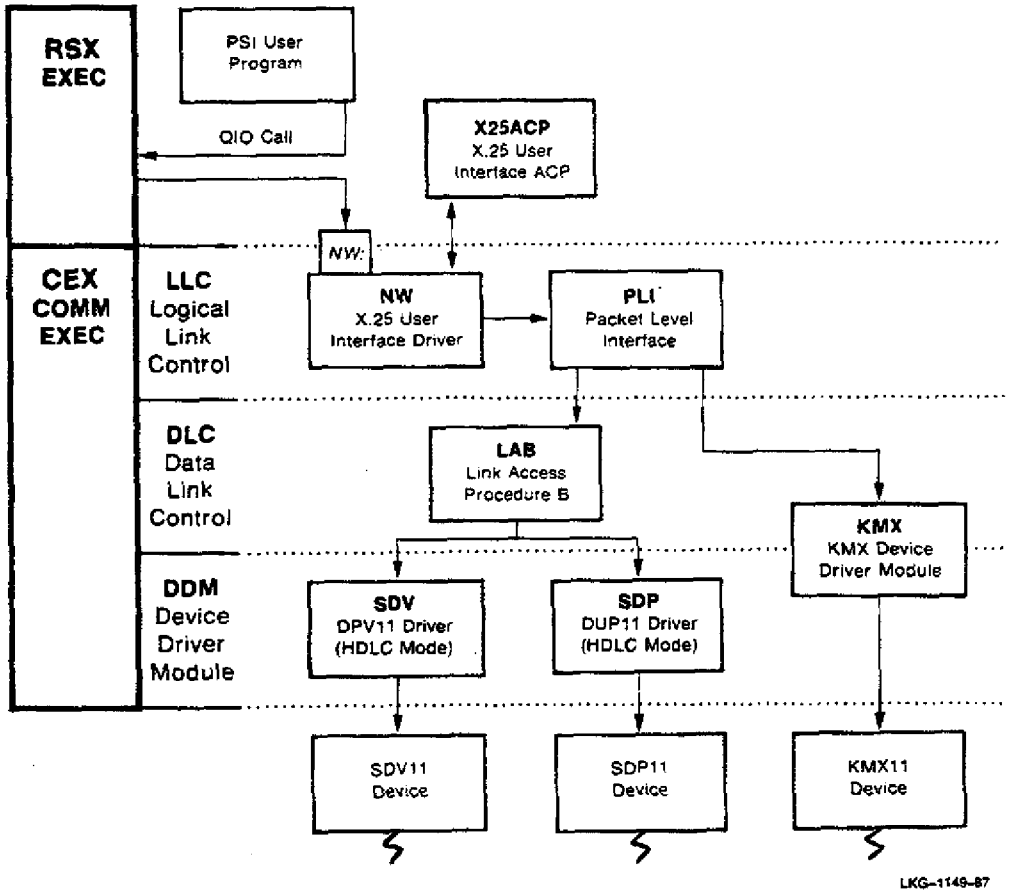


Figure 8–31 illustrates KMX, a PSI device. The KMX device contains microcode that performs the LAPB protocol in hardware. PLI connects directly to the KMX DDM. The illustration shows how the DDMs are connected to their corresponding hardware devices.

Figure 8–31: Physical Communications Devices for a PSI System



8.8.2 DECnet Interface to X.25

When DECnet-RSX is combined with PSI, DECnet-RSX nodes can communicate over a PSDN. The component that provides the interface between DECnet and PSI is called data link mapping (DLM).

Figure 8–32 illustrates a combined DECnet-RSX/PSI system with DLM. The left portion of the diagram, except DLM, is virtually the same as Figure 8–3, and the right portion is virtually the same as Figure 8–30.

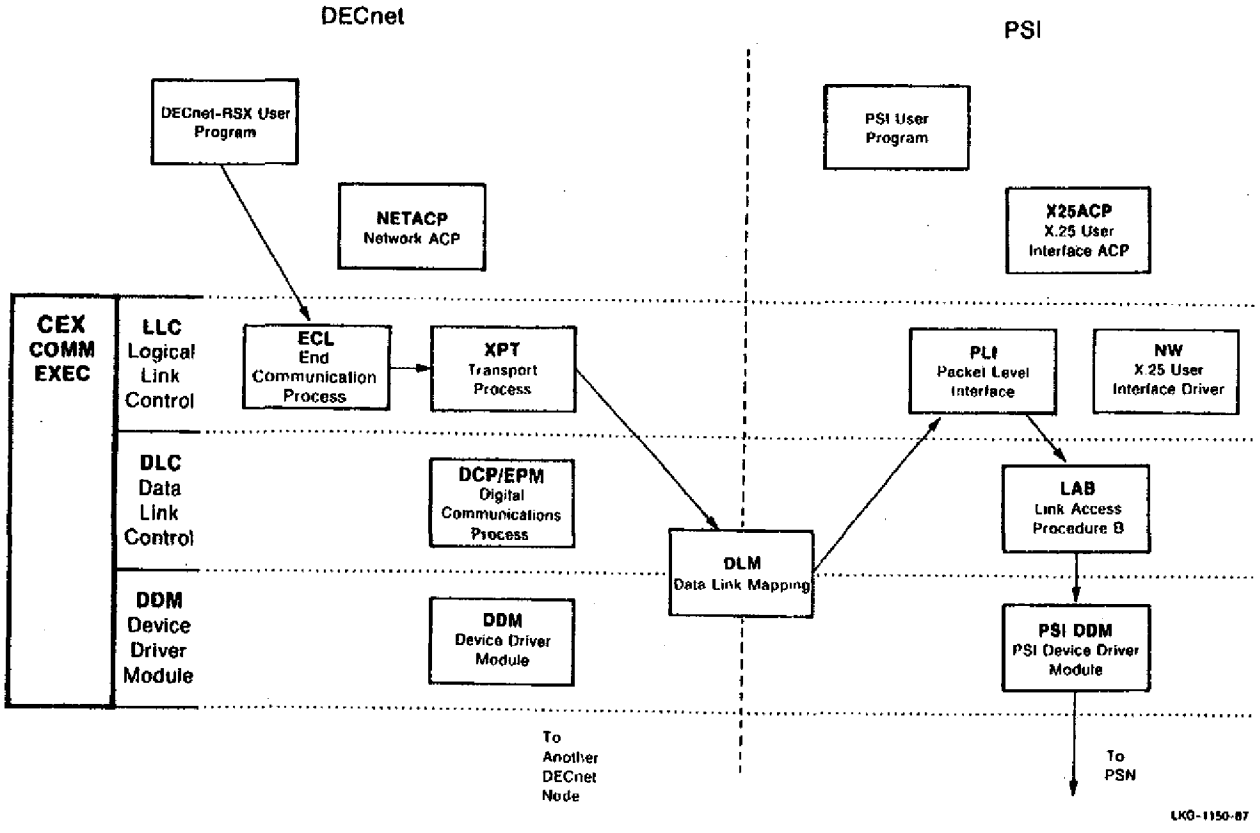


Figure 8–32: Combined DECnet-RSX/PSI System

Both the DECnet system and the PSI system can function independently. Each has its own user interface and its own lines. In the case of DECnet-RSX, the line goes to another DECnet node, and in the case of PSI, it goes to the PSDN.

When a DECnet program sends a message to a remote DECnet program over a PSDN, the following sequence occurs:

1. The program sends a QIO call to ECL.
2. ECL sends the message to XPT. XPT recognizes that the remote node can only be reached over the PSDN and sends the message to DLM.
3. DLM converts from the data format put out by ECL to the data format expected by PLI. DLM then sends the data to PLI.
4. PLI performs the processing of the X.25 level 3 protocol and sends the resulting packet to LAB.
5. LAB performs the X.25 level 2 processing and sends the packet to the appropriate PSI DDM.
6. The PSI DDM performs the X.25 level 1 processing that controls the physical device and sends the data over the line to the PSDN.

On the other side of the PSDN, exactly the reverse happens:

1. The PSI DDM, LAB, and PLI process the incoming data in turn.
2. PLI sends the processed data to DLM.
3. DLM converts the data from the format handled by PSI to the format handled by DECnet-RSX.
4. DLM sends this conversion to ECL. ECL processes the data and sends it to the DECnet user program.

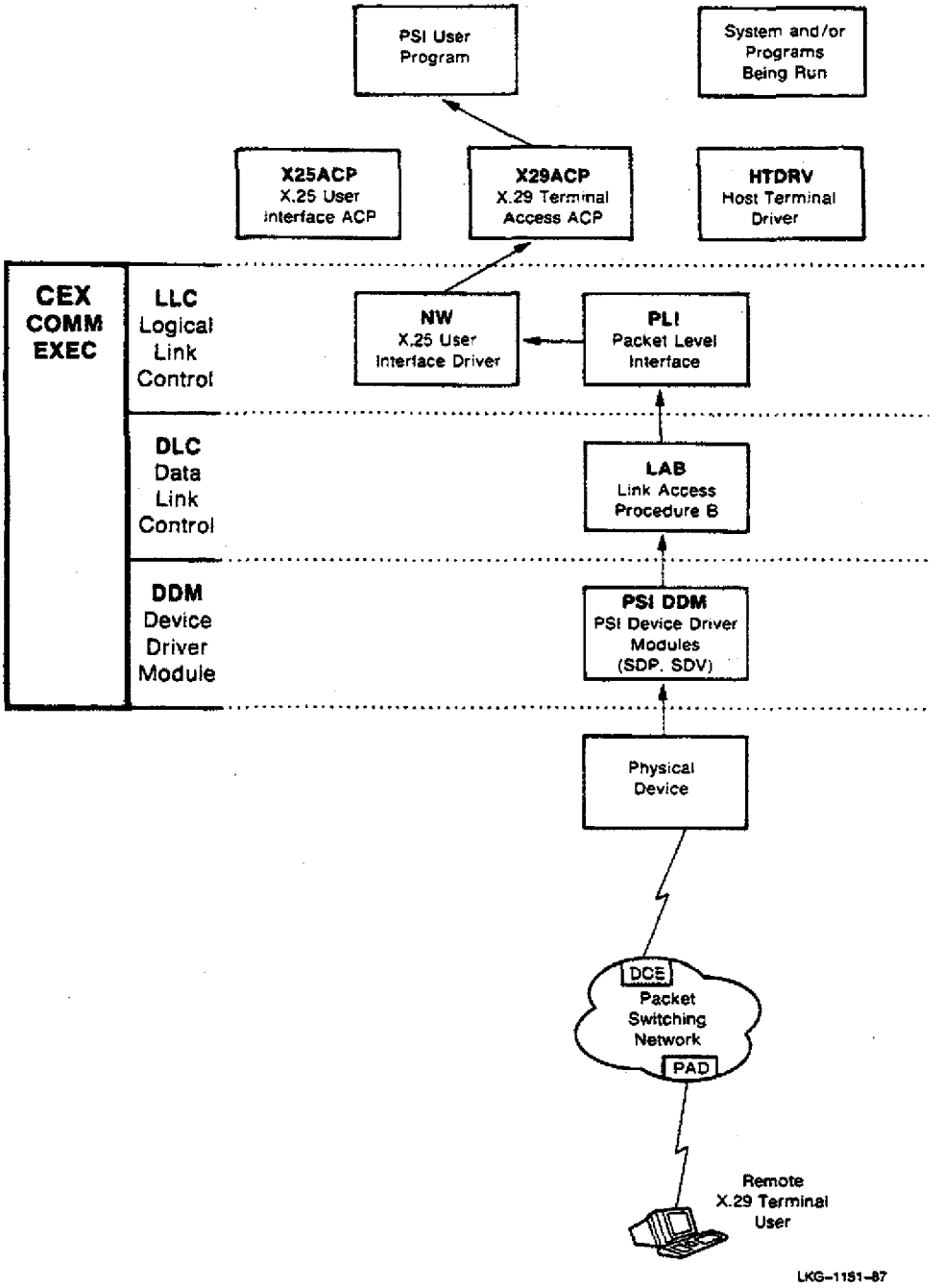
For both DECnet/PSI nodes, the data flow is two way.

8.8.3 X.29 Remote Terminals

An X.29 remote terminal allows a terminal without X.25 software to access either a PSI or DECnet-RSX/PSI system over a PSDN. Figure 8–33 illustrates a PSI system with an X.29 terminal.

The X.29 terminal user connects to a packet assembly/disassembly (PAD) facility of a PSDN and then specifies the address of the DTE to which the terminal is connected.

Figure 8–33: PSI System with an X.29 Terminal



The sequence of processing is:

1. The PAD collects characters entered at the terminal and converts them into packets. The packets are sent to the specified DTE and are transmitted through the physical device to the appropriate DDM, LAB, and PLI.
2. The DDM, LAB, and PLI handle the three protocols of X.25. The data is delivered to NW: (the pseudodevice), and NW: passes it to the appropriate user task.

In this case, the user task that handles the data is the X.29 terminal access ACP (X29ACP).

The terminal device (TT*n*:) for a local user is replaced on the host node by the host pseudoterminal device, HT*n*:. X29ACP assigns the flow of data to a given unit of the host terminal pseudodevice HT:, for example, HT0: or HT1:. This HT: device is associated with HTDRV. HTDRV communicates directly with the RSX operating system and/or given user programs.

The same pattern occurs for the reverse data flow. For example, if the program being run from the X.29 terminal sends output to the terminal, that output is sent through HTDRV, X29ACP, NW, PLI, LAB and the PSI DDM in turn. After being delivered to the DCE, the PSDN sends it to the PAD. PAD disassembles the packet and displays the characters on the terminal.

It is as if the remote X.29 terminal user were typing directly on a terminal on the local system. The user can make requests to the operating system or run one or more programs.

If PSI is running in this way, and if you included support for X.29 terminals during network generation, the remote X.29 terminal on a DECnet/PSI node has full access to the DECnet network functions.

For more information on PSI, see the *DECnet-RSX Network Generation and Installation Guide*.

8.9 Optional PSI Components

Optional PSI components include:

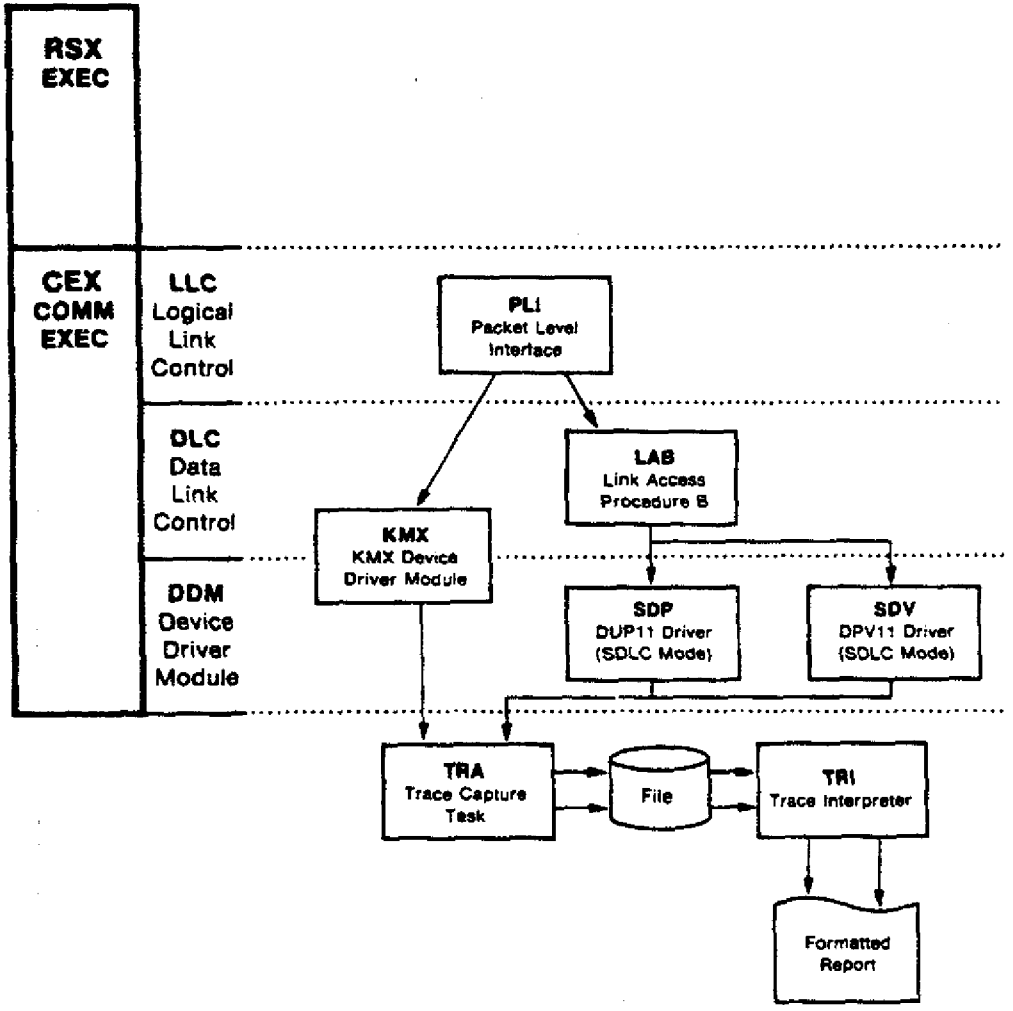
- PSI trace tasks
- KMX microcode tasks

8.9.1 PSI Trace Tasks

The trace capture task (TRA) and the trace interpreter task (TRI) assist in tracing line problems. Figure 8–34 illustrates these two tasks.

- TRA determines what types of packets are transmitted and received on a line, and monitors the line's performance.
- TRA collects snapshots of packets as they are passed from LAB to SDP and SDV.
- TRA collects snapshots of packets as they are passed from KMX.
- TRA writes this information to a file.
- TRI interprets and reads the file created by TRA and prints a file containing this information.

Figure 8–34: PSI Trace Tasks



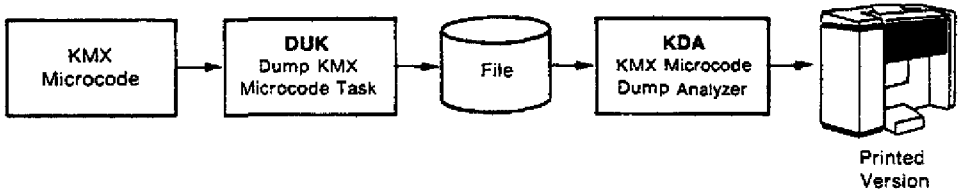
8.9.2 KMX Microcode Tasks

The KMX microcode dumper (DUK) and the KMS-11 microcode dump analyzer (KDA) are required components for KMX devices only. Figure 8–35 illustrates these two tasks.

- DUK dumps the microcode from the KMX device and writes it to a file.
- KDA then reads this file and produces a formatted listing of microcode instructions.

The PSI system provides these two tasks for dumping the KMX microcode and should only be used for unusual troubleshooting situations. They are intended for use by Digital software specialists. If you experience any problems with the KMX device, you should show the listing to a Digital software specialist.

Figure 8–35: KMX Microcode Tasks



LKG-1153-87

Appendixes

Insert tabbed
divider here.
Then discard
this sheet

A

CFE, NCP, and VNP Command Summary

This appendix summarizes the full command sets for the CFE, NCP, and VNP utilities.

Please review the graphic conventions outlined at the front of this manual, especially the use of braces { } and brackets [ ]. These conventions are used throughout the command summaries to specify parameter selection and optionality. The following list describes the shading and coloring conventions that are used in this appendix.

- Text that is shaded gray indicates commands or parameters that are valid for RSX–11 PSI users only.
- Text that is shaded red signifies commands or parameters that are valid for DECnet–11-PLUS systems only.
- In this appendix only, red ink is used to identify NCP commands and parameters that are RSX system specific. All CFE and VNP commands are RSX system specific.

A.1 CFE Command Summary

All CFE commands are privileged.

```
DEFINE {CIRCUIT circuit-id  }
       {KNOWN CIRCUITS      }

    CHANNEL channel-number
    COST cost
    COUNTER TIMER seconds
    DTE dte-address
    HELLO TIMER seconds
    LEVEL ONE COST cost
    LEVEL TWO COST cost
    MAXIMUM ROUTERS number
    MAXIMUM DATA byte-count
    MAXIMUM RECALLS retry-count
    MAXIMUM WINDOW block-count
    MULTIPOINT ACTIVE active-ratio
    NUMBER dte-address
    RECALL TIMER seconds
    ROUTER PRIORITY priority
    SERVICE {DISABLE}
            {ENABLE }
    STATE {OFF}
          {ON }
    TRIBUTARY trib-address
    USAGE {INCOMING}
          {OUTGOING}
```

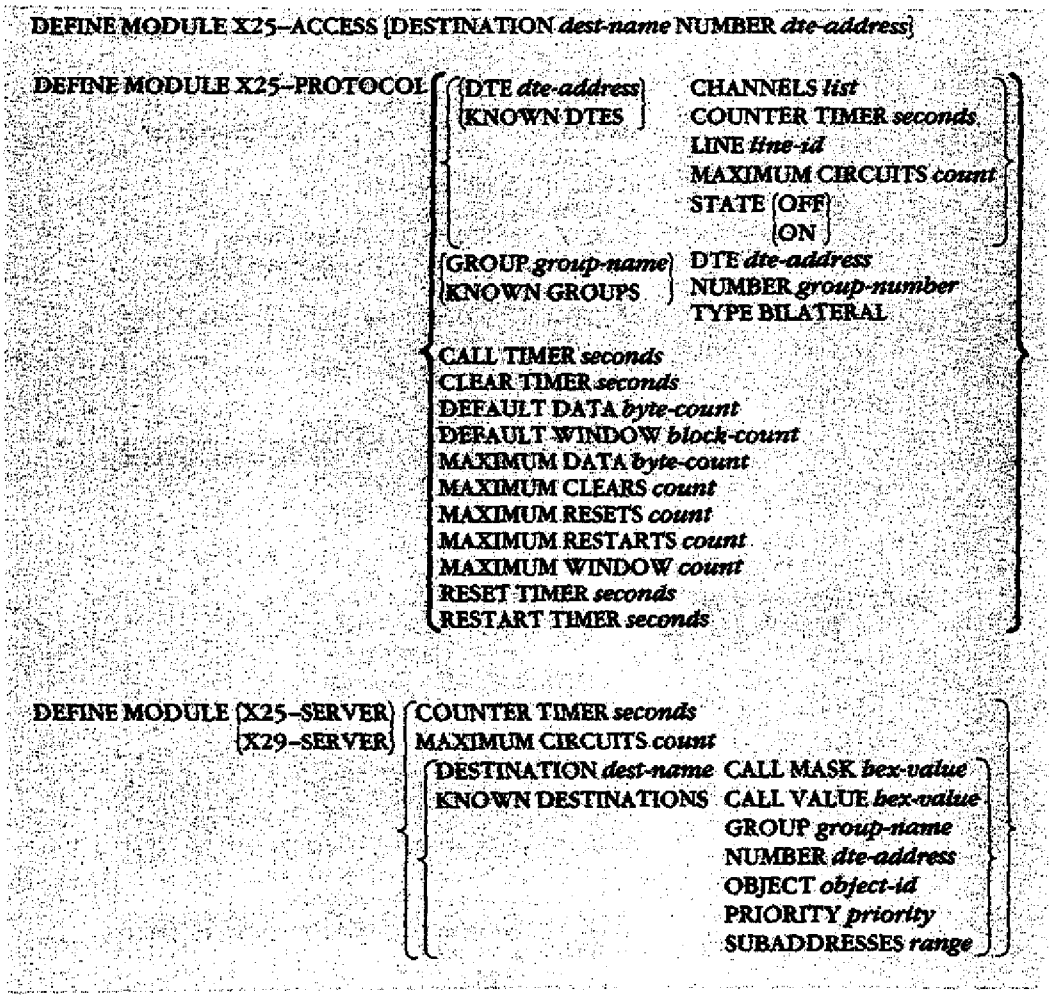
DEFINE EXECUTOR

ADDRESS *node-address*
AREA MAXIMUM COST *number*
AREA MAXIMUM HOPS *number*
BROADCAST ROUTING TIMER *seconds*
HOST *node-address*
IDENTIFICATION *id-string*
INACTIVITY TIMER *seconds*
INCOMING PROXY *control*
INCOMING TIMER *seconds*
MAXIMUM ADDRESS *node-address*
MAXIMUM AREAS *number*
MAXIMUM BROADCAST ENDNODES *number*
MAXIMUM COST *number*
MAXIMUM HOPS *number*
MAXIMUM LINKS *number*
MAXIMUM NODE COUNTERS *number*
MAXIMUM PATH SPLITS *number*
NAME *node-name*
OUTGOING PROXY *control*
OUTGOING TIMER *seconds*
RETRANSMIT FACTOR *number*
ROUTING TIMER *seconds*
SEGMENT BUFFER SIZE *number*
SUBADDRESSES *range*
VERIFICATION [STATE] { OFF | ON }

DEFINE { LINE *line-id* | KNOWN LINES }

[CONTROLLER] CSR *csr-address*
CONTROLLER { LOOPBACK | NORMAL }
COUNTER TIMER *seconds*
DUPLEX { FULL | HALF }
HOLDBACK TIMER *milliseconds*
MAXIMUM DATA *byte-count*
MAXIMUM RETRANSMITS *retry-count*
MAXIMUM WINDOW *block-count*
MULTIPOINT DEAD *dead-ratio*
PRIORITY *hardware-priority*
RETRANSMIT TIMER *milliseconds*
SPEED *baud-rate*
STATE { CLEARED | OFF | ON }
UNIT CSR *csr-address*
VECTOR *vector-address*

DEFINE
KNOWN LOGGING
LOGGING CONSOLE
LOGGING FILE
LOGGING MONITOR
EVENTS event-list
KNOWN EVENTS
STATE
OFF
ON


DEFINE MODULE X25-ACCESS [DESTINATION dest-name NUMBER dte-address]
DEFINE MODULE X25-PROTOCOL
DTE dte-address
KNOWN DTES
CHANNELS list
COUNTER TIMER seconds
LINE line-id
MAXIMUM CIRCUITS count
STATE
OFF
ON
GROUP group-name
KNOWN GROUPS
DTE dte-address
NUMBER group-number
TYPE BILATERAL
CALL TIMER seconds
CLEAR TIMER seconds
DEFAULT DATA byte-count
DEFAULT WINDOW block-count
MAXIMUM DATA byte-count
MAXIMUM CLEARS count
MAXIMUM RESETS count
MAXIMUM RESTARTS count
MAXIMUM WINDOW count
RESET TIMER seconds
RESTART TIMER seconds
DEFINE MODULE
X25-SERVER
X29-SERVER
COUNTER TIMER seconds
MAXIMUM CIRCUITS count
DESTINATION dest-name
KNOWN DESTINATIONS
CALL MASK hex-value
CALL VALUE hex-value
GROUP group-name
NUMBER dte-address
OBJECT object-id
PRIORITY priority
SUBADDRESSES range

DEFINE NODE *node-id*

DIAGNOSTIC FILE *file*
DUMP ADDRESS *address*
DUMP COUNT *number*
DUMP FILE *file*
HARDWARE ADDRESS *ethernet-address*
HOST *node-id*
LOAD FILE *file*
NAME *node-name*
SECONDARY [LOADER] *file*
SERVICE CIRCUIT *circuit-id*
SERVICE DEVICE *device-type*
SERVICE NODE VERSION {PHASE III | PHASE IV}
[SERVICE] PASSWORD *password*
TERTIARY [LOADER] *file*

DEFINE OBJECT *type-code*

COPIES {*number* | SINGLE}
NAME *object-name*
USER {DEFAULT | LOGIN}
VERIFICATION {INSPECT | OFF | ON}

DEFINE {PROCESS *process-name* | KNOWN PROCESSES}

MAXIMUM CONTROLLERS *count*
MAXIMUM LINES *number*
STATE {CLEARED | ON}

DEFINE SYSTEM

LARGE BUFFER SIZE *number*
[LOCATION] {FIRSTFIT | TOPDOWN}
MAXIMUM CONTROL BUFFERS *number*
MAXIMUM LARGE BUFFERS *number*
MAXIMUM SMALL BUFFERS *number*
MINIMUM RECEIVE BUFFERS *number*
POOL BYTE-AREA *byte-count*
POOL NAME *pool-name*
POOL PARTITION *partition-name*

EXIT [PURGE]

HELP [*command*] [*component-type*]

```
KILL

LIST (CIRCUIT circuit-id)
     (KNOWN CIRCUITS)

LIST EXECUTOR

LIST (LINE line-id)
     (KNOWN LINES)

LIST (KNOWN LOGGING)
     (LOGGING CONSOLE)
     (LOGGING FILE)
     (LOGGING MONITOR)

LIST MODULE X25-ACCESS (DESTINATION dest-name)
                       (KNOWN DESTINATIONS)

LIST MODULE X25-PROTOCOL (DTE dte-address)
                         (KNOWN DTES)
                         (GROUP group-name)
                         (KNOWN GROUPS)

LIST MODULE (X25-SERVER) [DESTINATION dest-name]
            (X29-SERVER) [KNOWN DESTINATIONS]

LIST (NODE node-id)
     (KNOWN NODES)

LIST (OBJECT type-code)
     (KNOWN OBJECTS)

LIST (PROCESS process-name)
     (KNOWN PROCESSES)

LIST SYSTEM

PURGE (CIRCUIT circuit-id) COUNTER TIMER
      (KNOWN CIRCUITS)

PURGE (LINE line-id) COUNTER TIMER
      (KNOWN LINES)
```

```
PURGE  { KNOWN LOGGING      }  { ALL EVENTS          }
       { LOGGING CONSOLE    }  { EVENTS event-list   }
       { LOGGING FILE       }  { KNOWN EVENTS        }
       { LOGGING MONITOR    }
```

```
PURGE MODULE X25-ACCESS  { DESTINATION dest-name }  ALL
                         { KNOWN DESTINATIONS    }

PURGE MODULE X25-PROTOCOL  { DTE dte-address }  { ALL           }
                           { KNOWN DTES      }  { COUNTER TIMER }
                           { GROUP group-name }  ALL
                           { KNOWN GROUPS     }

PURGE MODULE { X25-SERVER }  { COUNTER TIMER                }
             { X29-SERVER }  { DESTINATION dest-name ALL    }
                             { KNOWN DESTINATIONS ALL       }
```

```
PURGE { NODE node-id    }   ALL
      { KNOWN NODES     }   DIAGNOSTIC FILE
                            DUMP ADDRESS
                            DUMP COUNT
                            DUMP FILE
                            HARDWARE ADDRESS
                            HOST
                            LOAD FILE
                            SECONDARY [LOADER]
                            SERVICE CIRCUIT
                            SERVICE DEVICE
                            [SERVICE] PASSWORD
                            TERTIARY [LOADER]
```

```
PURGE { OBJECT object-type }
      { KNOWN OBJECTS      }
```

A.2 NCP Command Summary

NCP commands and descriptors that are supported by RSX–11S are summarized in Section A.3.

If you specify the parameter ALL, you cannot include any other parameters.

The following conventions are used in this section:

* = Command cannot be executed with the TELL prefix.
P = Privileged
NP = Nonprivileged
SCOPE is always privileged.

```
CLEAR { P ALL ALIASES          } [SCOPE] [GLOBAL                ]
      { NP ALIAS alias-name    }         [TERMINAL term-id      ]
      { NP KNOWN ALIASES       }
```

```
P  CLEAR {CIRCUIT circuit-id} COUNTER TIMER
         {KNOWN CIRCUITS    }
```

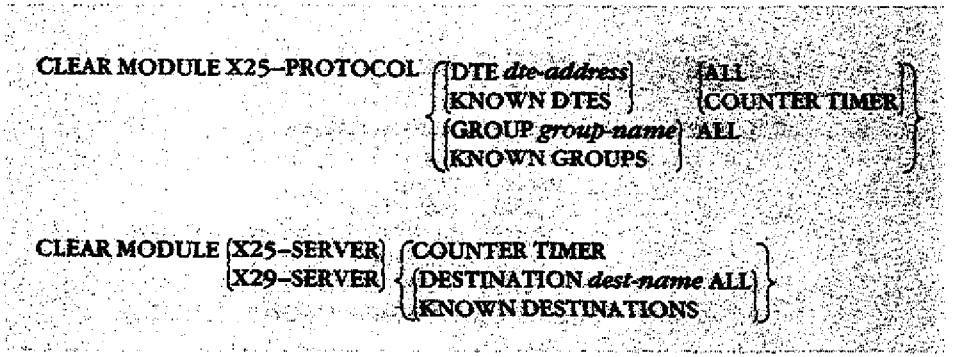
```
P  CLEAR EXECUTOR  HOST
                   RECEIVE PASSWORD
                   TRANSMIT PASSWORD
```

```
NP * CLEAR EXECUTOR NODE
```

```
P  CLEAR {LINE line-id  } {ALL           }
         {KNOWN LINES   } {COUNTER TIMER }
```

```
P  CLEAR {KNOWN LOGGING  } [NAME]          [CIRCUIT circuit-id                    ]
         {LOGGING CONSOLE} [EVENTS event-list] [LINE line-id                      ]
         {LOGGING FILE   } [KNOWN EVENTS    ] [MODULE {X25-PROTOCOL}              ]
         {LOGGING MONITOR}                    [        {X25-SERVER  }              ]
                                              [        {X29-SERVER  }              ]
                                              [NODE node-id                        ]
                                              [SINK {EXECUTOR  }                   ]
                                                    {NODE node-id}
```

```
P  CLEAR MODULE X25-ACCESS {ALL DESTINATIONS      } [SCOPE {GLOBAL          }]
                           {DESTINATION dest-name }       {TERMINAL term-id }
                           {KNOWN DESTINATIONS    }
```



P CLEAR NODE *node-id*
- ALL
- CIRCUIT
- DIAGNOSTIC FILE
- DUMP ADDRESS
- DUMP COUNT
- DUMP FILE
- HARDWARE ADDRESS
- HOST
- LOAD FILE
- NAME
- SECONDARY LOADER
- SERVICE CIRCUIT
- SERVICE DEVICE
- SERVICE PASSWORD
- TERTIARY LOADER

P CLEAR (OBJECT *type-code* | KNOWN OBJECTS) ALL

P CLEAR PROCESS *process-name*

P * CLEAR SYSTEM

P CLEAR TRACE (LINE *line-id* | ACTIVE LINES)

NP * EXIT

NP * HELP [*command*] [*component-type*]

P **RMX-DUMP LINE** *line-id* [**FILENAME** *file*]

P **LOAD NODE** *node-id* **ADDRESS** *node-address*
FROM *file*
HOST *node-id*
NAME *node-name*
PHYSICAL ADDRESS *ethernet-address*
SECONDARY [LOADER] *file*
SERVICE DEVICE *device-type*
SERVICE NODE VERSION {**PHASEIII** | **PHASEIV**}
[SERVICE] PASSWORD *password*
TERTIARY [LOADER] *file*
VIA *circuit-id*

P **LOAD VIA** *circuit-id* **ADDRESS** *node-address*
FROM *file*
HARDWARE ADDRESS *ethernet-address*
HOST *node-id*
NAME *node-name*
PHYSICAL ADDRESS *ethernet-address*
SECONDARY [LOADER] *file*
SERVICE DEVICE *device-type*
SERVICE NODE VERSION {**PHASE III** | **PHASE IV**}
[SERVICE] PASSWORD *password*
TERTIARY [LOADER] *file*

P **LOOP CIRCUIT** *circuit-id* [[**NODE** *node-name* | **PHYSICAL ADDRESS** *e-address*] [**HELP** { **FULL** | **RECEIVE** | **TRANSMIT** }] [**ASSISTANT NODE** *node-name* | **ASSISTANT PHYSICAL ADDRESS** *e-address*]] [**COUNT** *count* | **LENGTH** *length* | **WITH** { **MIXED** | **ONES** | **ZEROES** }]

NP **LOOP** {**NODE** *node-id* [*acc-con-info*] | **EXECUTOR**} **COUNT** *count*
LENGTH *length*
WITH {**MIXED** | **ONES** | **ZEROES**}

P LOOP LINE *line-id* COUNT *count*
LENGTH *length*
WITH {MIXED | ONES | ZEROES}

NP SET ALIAS *alias-name* DESTINATION *dest-node* [SCOPE] {GLOBAL | TERMINAL *term-id*}

P SET {CIRCUIT *circuit-id* | KNOWN CIRCUITS}
COST *cost*
COUNTER TIMER *seconds*
HELLO TIMER *seconds*
LEVEL 1 COST *cost*
LEVEL 2 COST *cost*
MULTIPOINT ACTIVE *active-ratio*
OWNER {DLX | XPT}
SERVICE {DISABLE | ENABLE}
STATE {OFF | ON | SERVICE}
TRIBUTARY *trib-address*

P SET EXECUTOR
HOST *node-id*
INACTIVITY TIMER *seconds*
INCOMING PROXY *control*
INCOMING TIMER *seconds*
MAXIMUM BROADCAST ENDNODES *number*
MAXIMUM PATH SPLITS *number*
OUTGOING PROXY *control*
OUTGOING TIMER *seconds*
RECEIVE PASSWORD *password*
RETRANSMIT FACTOR *number*
ROUTING TIMER *seconds*
SEGMENT BUFFER SIZE *number*
* STATE {OFF | ON [FIXED | UNFIXED] | SHUT}
SUBADDRESSES *range*
TRANSMIT PASSWORD *password*
VERIFICATION [STATE] {OFF | ON}

NP * SET EXECUTOR NODE *node-id* [*acc-con-info*]

P SET KNOWN LINES STATE {OFF / ON}

Loading options:

ALL
DEAD TIMER *milliseconds*
DELAY TIMER *milliseconds*
DUPLEX {FULL / HALF}
[LOCATION] {FIRSTFIT / TOPDOWN}

Loaded options:

COUNTER TIMER *seconds*
OWNER {DLX / PLI}

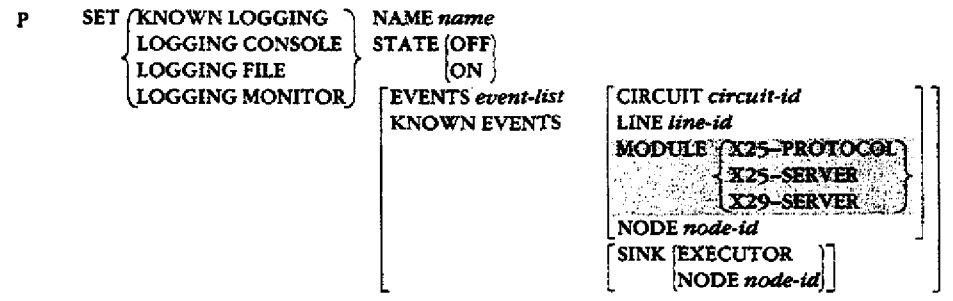
P SET LINE *line-id* STATE {OFF / ON}

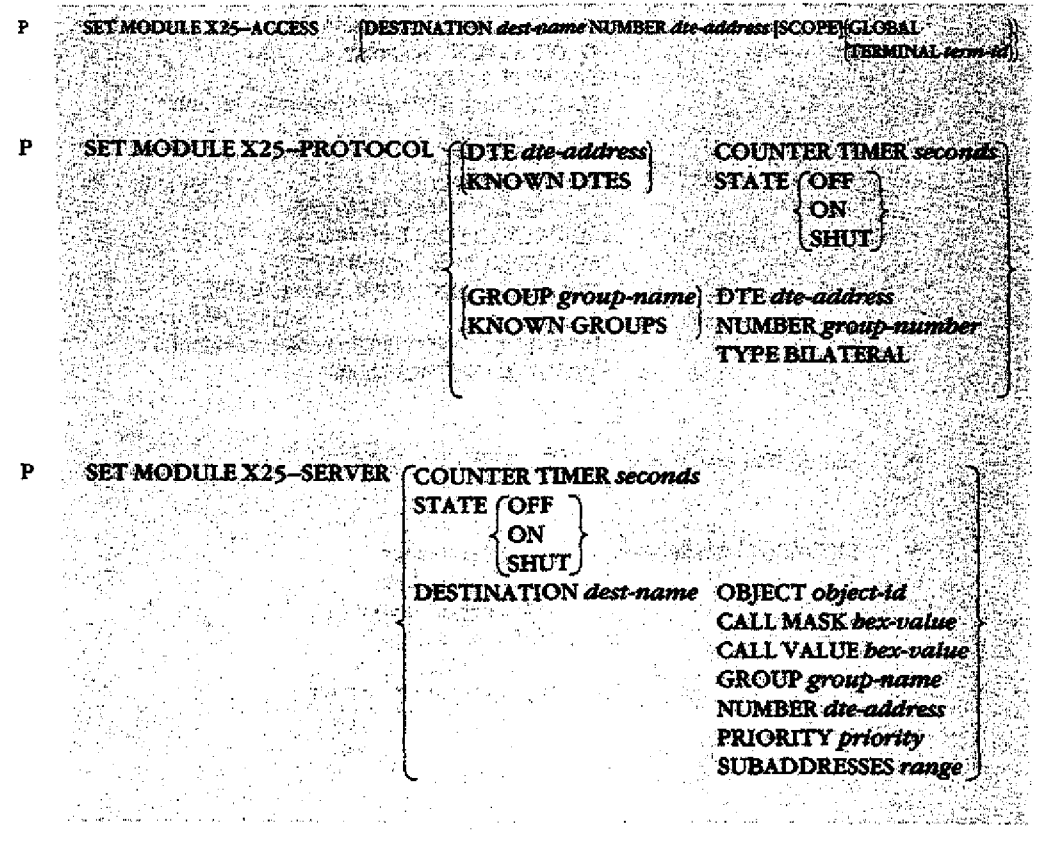
Loading options:

ALL
[CONTROLLER] CSR *csr-address*
DEAD TIMER *milliseconds*
DELAY TIMER *milliseconds*
DUPLEX {FULL / HALF}
[LOCATION] {FIRSTFIT / TOPDOWN}
MULTIPOINT DEAD *dead-ratio*
PRIORITY *hardware-priority*
UNIT CSR *csr-address*
VECTOR *vector-address*

Loaded options:

CONTROLLER {LOOPBACK / NORMAL}
COUNTER TIMER *seconds*
OWNER {DLX / PLI}


P
SET
KNOWN LOGGING
LOGGING CONSOLE
LOGGING FILE
LOGGING MONITOR
NAME name
STATE
OFF
ON
EVENTS event-list
KNOWN EVENTS
CIRCUIT circuit-id
LINE line-id
MODULE
X25-PROTOCOL
X25-SERVER
X29-SERVER
NODE node-id
SINK
EXECUTOR
NODE node-id


P
SET MODULE X25-ACCESS
DESTINATION dest-name NUMBER dte-address
SCOPE
GLOBAL
TERMINAL term-id
P
SET MODULE X25-PROTOCOL
DTE dte-address
KNOWN DTES
COUNTER TIMER seconds
STATE
OFF
ON
SHUT
GROUP group-name
KNOWN GROUPS
DTE dte-address
NUMBER group-number
TYPE BILATERAL
P
SET MODULE X25-SERVER
COUNTER TIMER seconds
STATE
OFF
ON
SHUT
DESTINATION dest-name
OBJECT object-id
CALL MASK hex-value
CALL VALUE hex-value
GROUP group-name
NUMBER dte-address
PRIORITY priority
SUBADDRESSES range

P SET MODULE X29-SERVER { COUNTER TIMER *seconds* | DESTINATION *dest-name* | OBJECT *object-id* | CALL MASK *hex-value* | CALL VALUE *hex-value* | GROUP *group-name* | NUMBER *dte-address* | PRIORITY *priority* | SUBADDRESSES *range* }

P SET NODE *node-id*
ADDRESS *node-address*
DIAGNOSTIC FILE *file*
DUMP ADDRESS *address*
DUMP COUNT *number*
DUMP FILE *file*
HARDWARE ADDRESS *ethernet-address*
HOST *node-id*
LOAD FILE *file*
NAME *node-name*
SECONDARY [LOADER] *file*
SERVICE CIRCUIT *circuit-id*
SERVICE DEVICE *device-type*
SERVICE NODE VERSION {PHASEIII | PHASEIV}
[SERVICE] PASSWORD *service-password*
TERTIARY [LOADER] *file*

P SET NODE *node-name* CIRCUIT *circuit-id*

P SET OBJECT *type-code*
COPIES {*number* | SINGLE}
NAME *object-name*
USER {DEFAULT | LOGIN}
VERIFICATION {INSPECT | OFF | ON}

P SET PROCESS *process-name* ALL
[LOCATION] {FIRSTFIT | TOPDOWN}
MAXIMUM CONTROLLERS *count*
MAXIMUM LINES *count*
PARTITION *partition-name*

P SET SYSTEM [TOP]

```
P   SET TRACE [LINE line-id      ]      BUFFER SIZE block-count
              [ACTIVE LINES     ]      FILE file
                                       STATE [OFF]
                                             [ON ]
```

```
NP  SHOW {ALL ALIASES       } [CHARACTERISTICS] [SCOPE] {GLOBAL           } [TO file]
         {ALIAS alias-name  } [SUMMARY        ]       {TERMINAL term-id  }
         {KNOWN ALIASES     }
```

```
NP  SHOW {AREA area-number    } [CHARACTERISTICS] [TO file]
         {ACTIVE AREAS        } [STATUS         ]
         {KNOWN AREAS         } [SUMMARY        ]
         {SIGNIFICANT AREAS   }
```

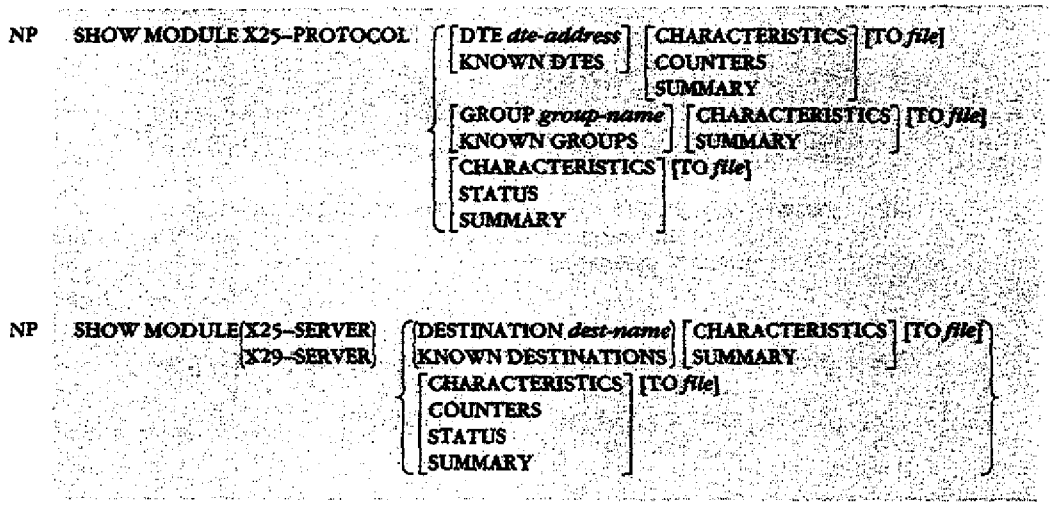
```
NP  SHOW {CIRCUIT circuit-id     } [CHARACTERISTICS] [TO file]
         {ACTIVE CIRCUITS        } [COUNTERS       ]
         {KNOWN CIRCUITS         } [STATUS         ]
         {SIGNIFICANT CIRCUITS   } [SUMMARY        ]
```

```
NP  SHOW EXECUTOR [CHARACTERISTICS] [TO file]
                  [COUNTERS       ]
                  [STATUS         ]
                  [SUMMARY        ]
```

```
NP  SHOW {LINE line-id      } [CHARACTERISTICS] [TO file]
         {ACTIVE LINES      } [COUNTERS       ]
         {KNOWN LINES       } [STATUS         ]
         {SIGNIFICANT LINES } [SUMMARY        ]
```

```
NP  SHOW {ACTIVE LOGGING      } [CHARACTERISTICS] [KNOWN SINKS    ] [TO file]
         {KNOWN LOGGING       } [EVENTS         ] [SINK NODE node-id]
         {LOGGING CONSOLE     } [STATUS         ]
         {LOGGING FILE        } [SUMMARY        ]
         {LOGGING MONITOR     }
         {SIGNIFICANT LOGGING }
```

```
NP  SHOW MODULE X25-ACCESS {ALL DESTINATIONS     } [CHARACTERISTICS] [SCOPE] [TO file]
                           {DESTINATION dest-name} [SUMMARY        ]
                           {KNOWN DESTINATIONS   }
```



NP SHOW { NODE *node-id* / ACTIVE NODES / ADJACENT NODES / KNOWN NODES / LOOP NODES / SIGNIFICANT NODES } [CHARACTERISTICS / COUNTERS / STATUS / SUMMARY] [TO *file*]

NP SHOW { OBJECT *type-code* / KNOWN OBJECTS } [CHARACTERISTICS / SUMMARY] [TO *file*]

NP SHOW { PROCESS *process-name* / ACTIVE PROCESSES / KNOWN PROCESSES } [STATUS / SUMMARY] [TO *file*]

NP SHOW SYSTEM [CHARACTERISTICS / COUNTERS / STATUS / SUMMARY] [TO *file*]

NP SHOW TRACE [STATUS / SUMMARY] [TO *file*]

NP TELL *node-id* [*acc-con-info*] *ncp-command*

P TRIGGER NODE *node-id* VIA *circuit-id*
PHYSICAL ADDRESS *ethernet-address*
[SERVICE] PASSWORD *service-password*

P TRIGGER VIA *circuit-id* PHYSICAL ADDRESS *ethernet-address*
[SERVICE] PASSWORD *service-password*

P ZERO {CIRCUIT *circuit-id* | KNOWN CIRCUITS} [COUNTERS]

P ZERO EXECUTOR [COUNTERS]

P ZERO {LINE *line-id* | KNOWN LINES} [COUNTERS]

P ZERO MODULE X25-PROTOCOL {DTE *dte-address* | KNOWN DTES} [COUNTERS]

P ZERO MODULE {X25-SERVER | X29-SERVER} [COUNTERS]

P ZERO {NODE *node-id* | KNOWN NODES} [COUNTERS]

P ZERO SYSTEM [COUNTERS]

A.3 NCP Command Summary for RSX–11S Systems

The following commands are supported by the RSX–11S NCP and the RSX–11S NICE. The privileged (P and nonprivileged (NP classifications apply only to commands sent from a remote system to an RSX–11S NICE that has been built with a privileged password. All commands can be initiated both locally and remotely unless one of the following restrictions is indicated:

? = Command cannot be initiated from a remote node.
! = Command cannot be executed by the RSX–11S NCP.

NP LOOP [NODE *node-id* [*acc-con-info*]] [COUNT *count*] [LENGTH *length*] [WITH [MIXED] [ONES] [ZEROS]]

P SET CIRCUIT *circuit-id* STATE [OFF] [ON]

P SET EXECUTOR HOST *node-id*

P ? SET LINE *line-id* CONTROLLER [LOOPBACK] [NORMAL]

P SET LOGGING CONSOLE STATE [OFF] [ON]

NP SHOW [CIRCUIT *circuit-id*] [! ACTIVE CIRCUITS] [! KNOWN CIRCUITS] [! SIGNIFICANT CIRCUITS] [COUNTERS] [STATUS] [SUMMARY]

NP SHOW EXECUTOR [CHARACTERISTICS] [COUNTERS] [STATUS] [SUMMARY]

NP SHOW [LINE *line-id*] [! ACTIVE LINES] [! KNOWN LINES] [! SIGNIFICANT LINES] [COUNTERS] [STATUS] [SUMMARY]

NP SHOW LOGGING CONSOLE STATUS

NP SHOW [NODE *node-id*] [! ACTIVE NODES] [! KNOWN NODES] [CHARACTERISTICS] [COUNTERS] [STATUS] [SUMMARY]

```
NP   SHOW SYSTEM [CHARACTERISTICS]
                  [COUNTERS      ]
                  [STATUS        ]
                  [SUMMARY       ]

P    ZERO { CIRCUIT circuit-id } [COUNTERS]
          {  EXECUTOR           }
          {  LINE line-id       }
          { ! NODE node-id      }
          {  SYSTEM             }

P    ! ZERO KNOWN {CIRCUITS} [COUNTERS]
                  {LINES   }
                  {NODES   }
```

A.4 VNP Command Summary

All VNP commands are privileged.

If you specify ALL, you cannot include any other parameters.

```
CLEAR { ALL ALIASES          } [SCOPE] { GLOBAL          }
      { ALIAS alias-name     }         { TERMINAL term-id }
      { KNOWN ALIASES        }
```

```
CLEAR EXECUTOR  HOST
                RECEIVE PASSWORD
                TRANSMIT PASSWORD
```

```
CLEAR { LINE line-id  } ALL
      { KNOWN LINES   }
```

```
CLEAR { KNOWN LOGGING   } [ [NAME]                                                    ]
      { LOGGING CONSOLE }   [ EVENTS event-list   [ LINE line-id                    ] ]
      { LOGGING FILE    }   [ KNOWN EVENTS        [ NODE node-id                    ] ]
      { LOGGING MONITOR }   [                     [ SINK { EXECUTOR          }   ] ]
                            [                            { NODE { node-id  }   ] ]
                            [                                   { $HOST    }   ] ]
```

```
CLEAR NODE node-id  NAME
```

```
CLEAR { OBJECT type-code } ALL
      { KNOWN OBJECTS     }
```

```
CLEAR PROCESS process-name
```

```
CLEAR SYSTEM
```

```
EXIT
```

```
HELP [command] [component-type]
```

```
SET ALIAS alias-name DESTINATION dest-node [SCOPE] { GLOBAL          }
                                                   { TERMINAL term-id }
```

```
SET [CIRCUIT circuit-id  ]   COST cost
    [KNOWN CIRCUITS      ]   HELLO TIMER seconds
                             LEVEL 1 COST cost
                             LEVEL 2 COST cost
                             MULTIPOINT ACTIVE active-ratio
                             OWNER [DLX    ]
                                   [XPT    ]
                             SERVICE [DISABLE]
                                     [ENABLE ]
                             STATE { OFF     }
                                   { ON      }
                                   { SERVICE }
                             TRIBUTARY trib-address
```

```
SET EXECUTOR                 HOST node-id
                             INACTIVITY TIMER seconds
                             INCOMING PROXY control
                             INCOMING TIMER seconds
                             MAXIMUM BROADCAST ENDNODES number
                             OUTGOING PROXY control
                             OUTGOING TIMER seconds
                             RECEIVE PASSWORD password
                             RETRANSMIT FACTOR number
                             ROUTING TIMER seconds
                             SEGMENT BUFFER SIZE number
                             STATE { OFF                }
                                   { ON [FIXED        ] }
                                   { ON [UNFIXED      ] }
                             TRANSMIT PASSWORD password
                             VERIFICATION [STATE] [OFF]
                                                  [ON ]
```

SET KNOWN LINES

Loading options:

```
ALL
DEAD TIMER milliseconds
DELAY TIMER milliseconds
DUPLEX [FULL]
       [HALF]
```

Loaded Option:

```
COUNTER TIMER seconds
```

SET LINE *line-id*

Loading options:

ALL
[CONTROLLER] CSR *csr-address*
DEAD TIMER *milliseconds*
DELAY TIMER *milliseconds*
DUPLEX {FULL | HALF}
MULTIPOINT DEAD *dead-ratio*
PRIORITY *hardware-priority*
UNIT CSR *csr-address*
VECTOR *vector-address*

SET {KNOWN LOGGING | LOGGING CONSOLE | LOGGING FILE | LOGGING MONITOR} NAME *name*
STATE {OFF | ON}
[EVENTS *event-list*]
[KNOWN EVENTS]
[LINE *line-id*]
[NODE *node-id*]
[SINK {EXECUTOR | NODE {*node-id* | $HOST}}]

SET MODULE X25-SERVER STATE {OFF | ON}

SET NODE *node-id* NAME *node-name*

SET NODE *node-name* CIRCUIT *circuit-id*

SET OBJECT *type-code* COPIES {*number* | SINGLE}
NAME *object-name*
USER {DEFAULT | LOGIN}
VERIFICATION {INSPECT | OFF | ON}

SET PROCESS *process-name* [ALL]
MAXIMUM CONTROLLERS *count*
MAXIMUM LINES *count*
PARTITION *partition-name*

SET SYSTEM

```
SHOW { ALL ALIASES                } [ CHARACTERISTICS ] [SCOPE] { GLOBAL                }
     { ALIAS alias-name           } [ SUMMARY         ]        { TERMINAL term-id       }
     { KNOWN ALIASES              }

SHOW { CIRCUIT circuit-id         } [ CHARACTERISTICS ]
     { KNOWN CIRCUITS             } [ SUMMARY         ]

SHOW EXECUTOR [ CHARACTERISTICS ]
              [ SUMMARY         ]

SHOW { LINE line-id               } [ CHARACTERISTICS ]
     { KNOWN LINES                } [ SUMMARY         ]

SHOW { KNOWN LOGGING              } [ CHARACTERISTICS ] [ KNOWN SINKS      ]
     { LOGGING CONSOLE            } [ EVENTS          ] [ SINK NODE node-id ]
     { LOGGING FILE               } [ SUMMARY         ]
     { LOGGING MONITOR            }

SHOW MODULE X25-ACCESS { ALL DESTINATIONS      } [ CHARACTERISTICS ] [SCOPE] { GLOBAL          }
                       { DESTINATION dest-name } [ SUMMARY         ]        { TERMINAL term-id }
                       { KNOWN DESTINATIONS    }

SHOW MODULE X25-PROTOCOL { [ DTE dte-address ] } [ CHARACTERISTICS ]
                         { [ KNOWN DTES      ] } [ SUMMARY         ]
                         { [ GROUP group-name ] }
                         { [ KNOWN GROUPS    ] }

SHOW MODULE { X25-SERVER } [ DESTINATION dest-name ] [ CHARACTERISTICS ]
            { X29-SERVER } [ KNOWN DESTINATIONS    ] [ SUMMARY         ]

SHOW { NODE node-id               } [ CHARACTERISTICS ]
     { ADJACENT NODES             } [ SUMMARY         ]
     { KNOWN NODES                }
     { LOOP NODES                 }

SHOW { OBJECT type-code           } [ CHARACTERISTICS ]
     { KNOWN OBJECTS              } [ SUMMARY         ]

SHOW { PROCESS process-name       } [SUMMARY]
     { KNOWN PROCESSES            }

SHOW SYSTEM [ CHARACTERISTICS ]
            [ SUMMARY         ]
```

B

Object Type Codes

Table B–1 defines valid object type code values and describes their process type for network management. The values are expressed as decimal byte values. Digital reserves the right to add object types or to make changes to the descriptor formats used by the object types.

Table B–1: Object Type Codes

Object Type Code	Process Type
0	General task, user program
1	File Access Listener—FAL/DAP, Version 1
2-4	Reserved for DECnet use
5	RSX–11M Task Control—Version 1
6-14	Reserved for DECnet use
15	RSX–11M Task Control—Version 2
16	TLK utility
17	File Access Listener—FAL/DAP, Version 4
18	Host Task Loader utility
19	Network Information and Control Exchange
20	RSTS/E media transfer program
21-22	Reserved for DECnet use

(continued on next page)

Table B–1 (Cont.): Object Type Codes

Object Type Code	Process Type
23	Network terminal handler
24	Reserved for DECnet use
25	Network management loopback mirror
26	Network management event receiver
27-28	Reserved for DECnet use
29	PHONE utility
30-38	Reserved for DECnet use
42	Heterogeneous terminal host
43-62	Reserved for DECnet use
63	RSX DECnet test tool
64-127	Reserved for DECnet use
128-255	Reserved for customer extensions

C

Network Management Event Logger Interface

The logging monitor interface from the DECnet event-logging facility provides a mechanism by which a user-written program can process network events. You must specify the name of the monitoring program and the events to be logged by using the NCP SET LOGGING command, as follows:

```
NCP> SET LOGGING MONITOR NAME name STATE ON
NCP> SET LOGGING MONITOR EVENTS event-list
```

where

name is the name of the program to receive the event information (default: MON...).

event-list identifies one or more event classes and types to be logged. See Section 2.6.1 for the event list format and see Appendix D for a list of event classes and types.

NOTE

The event monitor facility can be used in conjunction with event logging on the console or to a file.

If the monitor is inactive, it is invoked when the first event is about to be processed. Your program should open the network (OPN$) and use the GND$ directive to retrieve the event information from the mailbox. The mailbox is a message buffer that receives the event information. When events occur, the

I/O status block for the GND$ call contains the following on return from the call:

Word 0:	Byte 0 = IS.SUC/IE.DAO
	Byte 1 = NT.EVT(6)
Word 1:	Number of bytes of event information

The mailbox returned by GND$ contains the event information in the following format:

Word 0,1:	is the source program name in Radix–50. The source program is the program that sent the event information. EVC... is the name of the program used for locally generated events; EVR... is the name of the program used for remotely generated events.
Word 2-*n*:	is the event information given as a NICE protocol message. The format of the information can be found in the *DNA Network Management Functional Specification*.

After processing the event data, the program can keep the mailbox open and wait for further events, or it can exit and be reinvoked when the next event occurs.

D

Event Class and Type Summary

Events are summarized by class and type in this appendix. In general, event classes relate to specific layers of the DECnet architecture. Event types relate to specific events within an event class.

NOTE

Asterisks are used throughout this appendix to flag events and event classes that are not logged by DECnet-RSX.

D.1 Event Classes

Event logging supports the event classes listed in Table D–1. DECnet-RSX does not log events for classes marked with an asterisk. However, processed events in these classes from other remote nodes are logged.

Table D–1: Event Classes

Event Class	Description
0	Network Management layer
1 *	Application layer
2	Session Control layer
3	End Communications layer
4	Transport layer

Table D–1 (Cont.): Event Classes

Event Class	Description
5	Data Link layer
6 *	Physical Link layer
7-31 *	Reserved for other common classes
32-63 *	Reserved for RSTS systems
64-95	RSX system specific
96-127 *	Reserved for TOPS-20 systems
128-159 *	Reserved for VMS systems
160-479 *	Reserved for future use

D.2 Event Message Format

Event messages have the following format:

```
   Event type class.type event-text
Occurred dd-mon-yy hh:mm:ss on node address
   (node-name)
   entity-type entity-name
   data
```

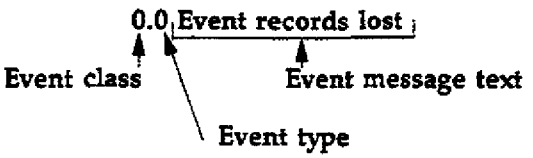
where

class	is the layer in which the event occurred.
type	is the specific type of event for that class.
event-text	is the text describing the event; does not print for RSX–11S.
address	is the address of the node at which the event occurred.
node-name	is the name of the node at which the event occurred.
dd-mon-yy	is the date: day, month, and year on which the event occurred.
hh:mm:ss	is the time: hour, minutes, and seconds at which the event occurred.
entity-type	is one of five types: area, line, circuit, node, or module. If an event is not associated with a particular entity, this line is not present. The entity types associated with each event are listed in Appendix F in Section F.4.
entity-name	is the name of the entity that caused the event.
data	is event-dependent text that gives more information about the event. Often this text includes the component type and name for which the event applies. It may also provide additional information about the cause of the event.

Example:

```
 Event type 0.3, Automatic service
Occurred 21-OCT-86 08:17:11 on node 19 (PITSBG)
Circuit UNA-0
Service type = Load
Status = Operation failure
```

The following sections list event messages by class and type for each layer. Individual events and entire event classes that are marked with an asterisk are not logged by DECnet-RSX components. Event messages shown in this appendix have the following format:



D.3 Network Management Layer Events

0.0 Event records lost

Events occurred too rapidly for the event logger to buffer them.

This message does not display any event qualifiers.

0.1 Automatic node counters*

A node counter timer expired.

This message displays the address and name of the node to which the event applies and the counters for that node.

0.3 Automatic service

An adjacent node requested an automatic circuit service operation.

This message displays the name of the circuit to which the event applies, as well as two event qualifiers: the service function performed (load or dump) and the status of the operation (requested, successful, or failed). If the operation fails, this status includes an error message from line watcher (LIN$$$) and details.

0.4 Line counters zeroed*

Line counters were zeroed.

This message displays the name of the line to which the event applies. The event logger records these counters prior to the execution of a request to zero them.

0.5 Node counters zeroed*

Node counters were zeroed.

This message displays the address and name of the node to which the event applies. The message can also include the node counters that were affected. The event logger records these counters prior to the execution of a request to zero them.

0.6 Passive loopback

The software initiated or terminated a passive loopback test on behalf of an adjacent node.

This message displays the circuit name to which the event applies and the state of operation qualifier, either initiated or terminated.

0.7 Aborted service request

An adjacent node requested a service over a circuit connected to the local node. However, a problem prevented it from being processed locally.

This message displays the name of the circuit to which the event applies and the reason for failure. Possible reasons are:

- **Circuit open error**

  LIN$$$ received a MOP message and was unable to acquire control of the circuit. Possible causes are: LIN$$$ did not have the privilege to perform the operation, or it could not set the substate of the circuit, or the circuit had another owner.

- **Circuit state change by higher level**

  The circuit was preempted by a higher priority function. For example, you used NCP to turn the line off.

- **Receive error**

  A line error occurred while trying to receive the request.

- **Receive timeout**

  The line message receive timer expired before the request could be received from the adjacent node. Possible causes are: the timer was too short, the line error level was too great for any message to get through, or the adjacent node stopped requesting.

- **Unrecognized request**

  A message was received but was not recognizable as a request for up-line dumping, down-line loading, or passive loopback testing. The adjacent node may be running an incompatible version of the line service protocol.

0.8 Automatic counters*

A counter timer for something other than a node or line expired (for example, a DTE counter timer).

0.9 Counters zeroed*

Counters for something other than a node or line were zeroed (for example, a DTE counter timer).

D.4 Session Control Layer Events

2.0 Local node state change

The operational state of the local node changed due to an operator command. Note that the transition from SHUT to OFF also happens automatically when the last logical link is disconnected under normal operation.

This message displays the reason for the state change, such as: operator command or normal operation, the old state (ON, OFF, SHUT, or RESTRICTED), and the new state (ON, OFF, SHUT, or RESTRICTED).

2.1 Access control failure

The local node rejected a connection request due to invalid access control information.

This message displays the name and address of the source node, the object type number and process ID of the source process requesting the connection, the object type number and process ID of the destination process to receive the connection request, and the invalid access control information (user ID, password, and account information).

D.5 End Communications Layer Events

3.0 Invalid message*

ECL received a message that could not be interpreted. This may indicate a software malfunction in either the local or the remote ECL.

This message displays the invalid ECL message. See the DNA Network Services Protocol Functional Specification for a description of ECL messages.

3.1 Invalid flow control*

The remote ECL attempted to invalidly modify the local flow control value. This may indicate a software malfunction in either the local or the remote ECL.

This message displays the invalid ECL message and the current flow control value. See the DNA Network Services Protocol Functional Specification for a description of flow control.

3.2 Node database reused

The local node received a connection request from or tried to initiate an outgoing connect to a node for which there is no counter block. All counter blocks have been previously used, and one of the previously used blocks is available for this new node. This results in the loss of node counters for the node that formerly occupied the database entry.

This message displays the address and name of the node for which the database entry was formerly used and the counters for that node.

D.6 Transport Layer Events

4.0 Aged packet loss

A packet has been discarded because it has visited too many nodes. This can be a normal occurrence when the network is reconfiguring its routing databases, or it can be a failure when the value specified for maximum hops is too small. This can cause the value specified for maximum visits to be too small for a path that should be usable.

This message displays the packet header only. This is information from the beginning of the packet. It consists of a hexadecimal byte of flags, the decimal destination and source node addresses, and a hexadecimal byte of forwarding data. See the DNA Routing Layer Functional Specification for additional information.

4.1 Node unreachable packet loss

A packet has been discarded because the local node found that the destination node was unreachable. This event provides a trace of what has happened to packets that are not reaching their destinations.

This message displays the packet header and the name of the circuit to which the event applies. The packet header is described in event 4.0.

4.2 Node out of range packet loss

A packet has been discarded because the destination node number was greater than the maximum node number known to the local node. Normally, this results from the addition of a new node to the network without increasing the value specified for maximum address on the local node and still expecting the local node to route packets to the new node.

This message displays the packet header and the name of the circuit to which the event applies. The packet header is described in event 4.0.

4.3 Oversized packet loss

A packet has been discarded because it was too large to forward to the appropriate adjacent node. Normally, this condition occurs because the adjacent node's buffer is too small or the source node sent a packet that was too large. The latter condition can be handled by setting a smaller segment size at the source node.

This message displays the packet header and the name of the circuit over which the packet was to be forwarded. The packet header is described in event 4.0.

4.4 Packet format error

A packet has been discarded due to a format error in the packet header. Usually, this results from a programming error in packet formatting by the adjacent node. It could also result from a circuit error not detected by circuit protocol.

This message displays a packet header and the name of the circuit to which the event applies. This consists of the first 6 bytes of the packet displayed in hexadecimal.

4.5 Partial routing update loss

A routing message containing node addresses greater than the maximum address known to the local node has been received. Subsequently, information on these nodes was lost. This occurs when the value specified for the maximum address on an adjacent node is increased but the same value for the local node is not increased.

This message displays the name of the circuit over which this message was received, the packet header (see event 4.0), and the highest node address in the routing update that was lost.

4.6 Verification reject

Initialization with an adjacent node has failed. The local node received an invalid password. The receive password does not match the remote node's transmit password. You must either change one of the passwords or clear the receive password.

This message displays the name of the circuit to which the event applies and the address and name of the adjacent node that failed to initialize.

4.7 Circuit down - circuit fault

Line and/or device hardware failure.

This message displays the name of the circuit to which the event applies and the reason for the event. Possible reasons are:

- **Adjacent node address change**

  The adjacent node changed addresses without going through the normal initialization sequence. This is logged when an adjacent node attempts to initialize with the local node but the adjacent node's address is not in the database.

- **Adjacent node address out of range**

  The adjacent node's address is greater than the maximum address defined for the local node. This can be caused by an incorrectly defined node address or by a failure to increase the maximum node address in the local node's database when a new node was added.

- **Adjacent node block size too small**

  The line block size provided by the adjacent node is too small for normal network operation. The block size may be set incorrectly at the adjacent node.

- **Adjacent node listener receive timeout**

  The node has received no message over the data link within the last 30 seconds. Most likely the remote node is not running.

- **Adjacent node listener received invalid data**

  A test message sent by the adjacent node contained invalid or corrupted data. This is most likely caused by a hardware problem.

- **Circuit synchronization lost**

  The normal circuit protocol was restarted or terminated by the adjacent node. Either a circuit exceeded an error threshold or network management initiated a circuit state change.

- **Data errors**

  The circuit was declared down by the local node's circuit protocol handler when the circuit exceeded an error threshold.

- **Invalid verification seed value**

  A transport initialization message sent by an adjacent node is improperly formatted. This is most likely caused by a remote network software problem.

- **Routing update checksum error**

  A routing update packet failed its internal integrity test.

- **Unexpected packet type**

  A packet was received out of the normal protocol sequence. For example, the local node received a normal data packet when it expected a verification packet.

- **Verification receive timeout**

  A required verification packet was not received from the adjacent node within the required response time. Either packets were lost on the circuit or a failure occurred at the adjacent node.

- **Version skew**

  The routing version of the adjacent node is unacceptable to the local node. The operator may have installed incorrect software at the adjacent node.

4.8 Circuit down

A software error occurred on the circuit and the circuit has been taken out of service. Common causes of this event include the following:

- There is a hardware problem with the line and/or a device.
- The remote circuit has recycled. This could result from an incorrect password sent from this node during initialization.
- The operator turned the remote circuit off and on.

This message displays the name of the circuit to which the event applies, the packet header described under event 4.0, and the reason for the event. See the reasons listed for event 4.7.

4.9 Circuit down - operator initiated

An operator has turned the circuit off.

This message displays the name of the circuit to which the event applies, the packet header described under event 4.0, the expected node address and name, and the reason for the event. See the reasons listed for the event in 4.7.

4.10 Circuit up

A remote node has been initialized on one of the circuits connected to the local node.

This message displays the name of the circuit to which the event applies, as well as the name and address of the newly initialized, adjacent node.

NOTE

On a routing node, this event does not imply that the node is reachable. Reachability is determined by the higher level routing algorithms. For the UNA and QNA, the event signifies that all basic initialization has been completed by the device and transceiver.

4.11 Initialization failure - line fault

A remote node failed to initialize with the local node due to a line error.

This message displays the name of the circuit to which the event applies and the reason for the event. See the reasons listed for the event in 4.7.

4.12 Initialization failure - software fault

A remote node failed to initialize with the local node due to a software error.

This message displays the name of the circuit to which the event applies, the packet header described under event 4.0, and the reason for the event. See the reasons listed for event 4.7.

4.13 Initialization failure - operator fault

A remote node failed to initialize with the local node due to an operator error.

This message displays the name of the circuit to which the event applies, the packet header described under event 4.0, the reason for the event listed for event 4.7, and the version received from the adjacent node.

4.14 Node reachability change*

There has been a change in the reachability of a particular node.

This message displays the new node status.

4.15 Adjacency up

For broadcast circuits, UNA and QNA, initialization has occurred with another node on the Ethernet. End nodes log this message for only one node.

This message displays the adjacent node number.

4.16 Adjacency rejected

For broadcast circuits, UNA and QNA, initialization has been rejected. The number of broadcast end nodes may have been exceeded.

The message displays the reason for the rejection.

4.18 Adjacency down

The remote node has recycled. This event could result from a remote node restart or an invalid protocol message.

The message displays the reason that the adjacent node is down.

4.19 Adjacency down - operator initiated*

The operator has turned the Ethernet circuit off. For DECnet-RSX, this event is logged as event 4.9. See 4.9.

D.7 Data Link Layer Events

5.0 Locally initiated state change*

An operator command changed the circuit state.

This message displays the name of the circuit to which the event applies, the old DDCMP state (HALTED, ISTRT, ASTRT, RUNNING, or MAINTENANCE), and the new DDCMP state. See the old DDCMP states. See the DNA DDCMP Functional Specification for a description of these states.

5.1 Remotely initiated state change*

A remote user changed the circuit state.

This message displays the name of the circuit to which the event applies and the old and new DDCMP state. See event 5.0 for the states.

5.2 Protocol restart received in maintenance mode*

The remote node restarted normal operation while the local node had the circuit in maintenance mode.

This message displays the name of the circuit to which the event applies.

5.3 Send error threshold*

Too many data transmission errors occurred.

This message displays the name of the circuit to which the event applies.

5.4 Receive error threshold*

Too many data reception errors occurred.

This message displays the name of the circuit to which the event applies.

5.5 Select error threshold*

Too many selection errors occurred.

This message displays the name of the circuit to which the event applies.

5.6 Block header format error*

DDCMP received an invalid block header.

This message displays the name of the circuit to which the event applies.

5.7 Selection address error*

The wrong tributary responded to the polling process. Only multipoint control stations experience this event. This event occurs when one of these stations receives a message that does not match the address of the currently selected tributary.

This message displays the name of the circuit to which the event applies.

5.8 Streaming tributary*

A tributary on the circuit is impeding the use of that circuit.

This message displays the name of the circuit to which the event applies.

5.9 Local buffer too small*

A local buffer is too small to receive a block of data.

This message displays the name of the circuit to which the event applies.

5.10 Restart*

The DTE completed a restart procedure. That is, X.25 level 3 either has sent a restart confirm signal to a received restart command or has received a restart confirm signal to a transmitted restart command.

This message displays the module name, the DTE name, the cause, and a diagnostic.

5.11 State change*

The operational state of a module was changed by the operator command.

This message displays the name of the module whose state was changed, the name of the DTE where the module resides, the reason for the change, the old state, and the new state.

5.12 Retransmit maximum exceeded*

The retry count for the restart retransmission expired.

This message displays the name of the module whose retransmit maximum was exceeded, the name of the DTE where the module resides, and the parameter for which the maximum was set.

5.13 Initialization failure

The line could not be initialized because of a device error.

This message displays the name of the device that could not be enabled. Depending upon the implementation, a reason for the failure might also be reported.

5.14 Send failed

An error occurred during a transmit operation.

This message displays the name of the line over which the transmit operation was attempted and the reason for failure. Possible reasons are:

- **Carrier check failed**

  The data link did not sense the receive signal that must accompany a transmit message. There is a failure in either the transmitting or the receiving hardware, possibly with the transceiver or its cable.

- **Excessive collisions**

  The maximum number of retransmissions due to collisions has been exceeded. This is caused when too many systems on the Ethernet are trying to transmit at once.

- **Frame too long***

  The transmit message was truncated. Either the local node tried to send a message that exceeded the maximum allowable size or an error in the local hardware cut off transmission before the actual maximum size was reached.

- **Open circuit**

  There is a break in the Ethernet coaxial cable. The event message will include an estimated distance to the failure in bit times. If other systems are reporting the same problem, the failure is on the network cable rather than with local hardware.

- **Remote failure to defer**

A remote system began transmitting after the allowable time for collisions had elapsed. This could indicate a weak transmitter on the local node or a problem with the remote carrier sense circuitry.

- **Short circuit**

  Either there is a short circuit in the Ethernet coaxial cable, or the transceiver or transceiver/controller cable failed. The event message will include an estimated distance to the failure in bit times. If other systems are reporting the same problem, the failure is on the network rather than with local hardware.

5.15 Receive failed

An error occurred during a receive operation.

This message displays the name of the line over which the receive operation was attempted and the reason for failure. Possible reasons are:

- **Block check error**

  The message failed the cyclic redundancy check (CRC). Possible causes are electromagnetic interference, late collisions, or improperly set hardware parameters, such as receiver squelch.

- **Data overrun**

  The message was lost due to a hardware failure such as insufficient hardware buffers or insufficient CPU time.

- **Frame too long**

  The message was discarded because a remote system sent a message that exceeded the maximum allowable length.

- **Framing error**

  The message did not contain a multiple of 8-bit bytes. Possible causes are electromagnetic interference, late collisions, or improperly set hardware parameters, such as receiver squelch.

- **System buffer unavailable**

  The message was discarded because there was no system buffer available to receive it. There are not enough system buffers on the local system.

- **Unrecognized frame destination***

The message was discarded because there was no active user with the specified protocol type or address enabled. Either the local system has not enabled a protocol or an address that it should have, or a remote system is trying to use a protocol that is locally unsupported.

- **User buffer unavailable***

  The message was discarded because there was no user buffer available to receive it. The user has not supplied enough buffers in the user process.

D.8 Physical Link Layer Events

6.0 Data set ready transition*

A transition in the data set ready signal was detected by the interface hardware.

The message displays the name of the line on which the transition occurred.

6.1 Ring indicator transition*

A transition in the ring indicator signal was detected by the interface hardware.

This message displays the name of the line on which the transition occurred.

6.2 Unexpected carrier transition*

A transition in the carrier signal was detected by the interface hardware.

This message displays the name of the line on which the transition occurred.

6.3 Memory access error*

The interface hardware tried to access nonexistent memory.

This message displays the name of the line on which the error occurred.

6.4 Communications interface error*

The interface hardware detected an error in itself.

This message displays the name of the line on which the error occurred.

6.5 Performance error*

A soft error has been detected. This error may degrade performance.

This message displays the name or the line on which the error was detected.

D.9 RSX System-Specific Events

64.1 Routing database corrupt*

The routing algorithm detected an inconsistency in the routing databases. The algorithm will not send routing messages.

This message displays the event text only.

64.2 Routing database restored*

The routing database has been restored by the receipt of valid routing messages from other nodes. The routing algorithm will start sending routing messages again.

This message displays the event text only.

68.14 Normal usage terminated*

The operator turned off the circuit.

This message displays the name of the circuit that was shut down.

93.0 State change*

This message displays the reason for the X.25 server module state change, the old state, and the new state.

94.0 DCE detected packet error*

The DCE detected an X.25 level 3 error on a circuit.

This message displays the name of the DTE associated with that circuit, the diagnostic code received from the network, and the first 6 bytes of the diagnostic packet.

E

Network Counter Summary

The network software maintains counters for lines, circuits, certain modules, nodes, and the system. In some cases, the counters respond to and reflect network events. Events are defined in the discussion of event logging in Section 2.6 In other cases, the counters respond to and reflect normal activities such as messages sent and messages received. Individual counter descriptions indicate whether that counter is incremented when a corresponding event occurs. See Appendix D for a description of event messages. Where possible, the reasons why each of the counters might be incremented are included in the description.

In this appendix, counters are presented in alphabetical order within component groups. If you receive a counter number instead of the usual text when executing commands remotely at an RSX node from a non-RSX node, refer to the lists that correlate counter type numbers with counter text in Appendix F. Then look up the counter description in this appendix for full details.

A counter does not display a value greater than its maximum value. When a counter overflows, it locks on the overflow value until it is zeroed. Counter displays with an angle bracket > indicate that the counter has overflowed. For example, if the maximum value for a counter is 254, its overflow display is >254. Each of the counter descriptions in this appendix includes the maximum value of the counter.

Each category of counters maintains a timing counter, measuring in seconds, since last zeroed. The timing counter is zeroed when its associated counters are zeroed and starts when they start. In this way, the timing counter logs the seconds since its associated counters were zeroed to provide a time frame for them. For example, if you examined the system counters and found that the control buffer allocation failed counter registered 700, you can determine the

frequency of the failures by also checking the associated seconds since last zeroed counter.

The counter descriptions in this appendix explain some uses of the counters, but they should not be considered an exhaustive trouble-shooting guide. For detailed information on counters or the software design and algorithms they represent, consult the various architectural specifications.

E.1 Circuit Counters

There are five groups of circuit counters, including one group in the Network Management layer, one group in the Transport layer, and three groups in the Data Link layer. The Data Link groups cover DDCMP, Ethernet, and X.25 permanent virtual circuits (PVCs).

E.1.1 Network Management Layer: All Circuits

Seconds since last zeroed

This counter starts when the other circuit counters are zeroed. It increments by 2 every 2 seconds. This counter is zeroed when the other circuit counters are zeroed, so as to provide a time frame for them. The overflow value is 65,534.

E.1.2 Transport Layer: All Circuits

Adjacency down

This counter records the number of times that a connection to an adjacent node was lost. The overflow value is 65,534.

Circuit down

This counter records the number of times that a circuit was declared down by the executor. The overflow value is 254.

Corruption loss

This counter records the number of times that a checksum failure occurred on a PSN. The overflow value is 254.

Initialization failure

This counter increments when the circuit could not be initialized by the executor for network use. The overflow value is 254.

Originating packets sent

This counter records the number of packets sent by the executor over the circuit. The overflow value is 4,294,967,294.

Peak Adjacencies

This counter indicates the maximum number of simultaneous adjacent nodes. The overflow value is 65,534.

Terminating packets received

This counter records the number of packets received by the executor with the executor as the destination. The overflow value is 4,294,967,294.

Transit congestion loss

This counter records the number of packets received by the executor that were to be routed to another node but were discarded because of heavy traffic on the output circuit. The overflow value is 65,534. This counter is kept for routing nodes only.

Transit packets received

This counter increments when the executor receives a packet that is to be routed to another node. The overflow value is 4,294,967,294. This counter is kept for routing nodes only.

Transit packets sent

This counter increments when the executor sends a packet through to another node. The overflow value is 4,294,967,294. This counter is kept for routing nodes only.

Bytes received

This counter increments when a data byte is received on the circuit. The count does not include Data Link Protocol overhead or bytes retransmitted by the Data Link layer. It can be used with the counter or data blocks received to determine the inbound traffic load on the circuit. The overflow value is 4,294,967,294.

Bytes sent

This counter increments when a data byte is sent on the circuit. The count does not include Data Link Protocol overhead or bytes retransmitted by the Data Link layer. It can be used with the counter or data blocks received to determine the outbound traffic load on the circuit. The overflow value is 4,294,967,294.

Data blocks received

This counter increments when a data block is received on the circuit. The count does not include Data Link Protocol overhead. This counter can be used as a statistical base for evaluating the other Data Link layer counters. The overflow value is 4,294,967,294.

Data blocks sent

This counter increments when a data block is sent on the circuit. The count does not include Data Link Protocol overhead or blocks retransmitted by the Data Link layer. It can be used as a statistical base for evaluating the other Data Link layer counters. The overflow value is 4,294,967,294.

Data errors inbound

This counter indicates the number of incoming data errors on the circuit. It can have any of the following qualifiers: negative acknowledgments (NAKs) sent, data field block check error; NAKs sent, reply response. The overflow value is 254.

Data errors outbound

This counter indicates the number of outgoing data errors on the circuit. It can have any of the following qualifiers: NAKs received, header block check error; NAKs received, data field block check error; NAKs received, reply response. The overflow value is 254.

Local buffer errors

This counter increments when a negative acknowledgment (NAK) is sent. It can have any of the following qualifiers: NAKs sent, buffer unavailable; NAKs sent, buffer too small. The overflow value is 254.

Local reply timeouts

This counter increments each time that a message is retransmitted because the retry timer for a sent message expired before an ACK was received from the remote node. The overflow value is 254.

Remote buffer errors

This counter increments when a negative acknowledgment (NAK) is received. It can have any of the following qualifiers: NAKs received, buffer unavailable; NAKs received, buffer too small. The overflow value is 254.

Remote reply timeouts

This counter increments each time that a message is retransmitted because the retry timer for a sent message expired before an ACK from your node was received at the remote node. The overflow value is 254.

Selection intervals elapsed

This counter records the number of times that the executor turned a circuit around or selected an adjacent node on both half duplex and multipoint circuits. This counter is used as a statistical base for the evaluation of the counter for selection timeouts. The overflow value is 65,534.

Selection timeouts

This counter records the number of times that the executor turned a circuit around or selected an adjacent node on both half duplex and multipoint circuits but the adjacent node failed to respond within the required time. This can be caused by blocks being lost on the circuit in either direction or by too small a value being specified for the executor's response timer. Blocks are usually lost because of a partial, temporary, or total failure of the communications line. This counter can have the following qualifier: no reply to select. The overflow value is 254.

Bytes received

This counter increments when a data byte is received on the circuit. The count does not include Data Link Protocol overhead or bytes retransmitted by the Data Link layer. It can be used with the counter or data blocks received to determine the inbound traffic load on the circuit. The overflow value is 4,294,967,294.

Bytes sent

This counter increments when a data byte is sent on the circuit. The count does not include Data Link Protocol overhead or bytes retransmitted by the Data Link layer. It can be used with the counter or data blocks received to determine the outbound traffic load on the circuit. The overflow value is 4,294,967,294.

Data blocks received

This counter increments when a data block is received on the circuit. The count does not include Data Link Protocol overhead. This counter can be used as a statistical base for evaluating the other Data Link layer counters. The overflow value is 4,294,967,294.

Data blocks sent

This counter increments when a data block is sent on the circuit. The count does not include Data Link Protocol overhead or blocks retransmitted by the Data Link layer. It can be used as a statistical base for evaluating the other Data Link layer counters. The overflow value is 4,294,967,294.

User buffer unavailable

This counter indicates the total number of times that no user buffer was available for an incoming frame that passed all filtering. User buffers are supplied by users on receive requests. The overflow value is 65,534.

Bytes received

This counter indicates the number of bytes of data received over the circuit since the counter was last zeroed. The overflow value is 4,294,967,294.

Bytes sent

This counter indicates the number of bytes of data sent over the circuit since the counter was last zeroed. The overflow value is 4,294,967,294.

Data blocks received

This counter indicates the number of data blocks received over the circuit since the counter was last zeroed. The overflow value is 4,294,967,294.

Data blocks sent

This counter indicates the number of data blocks sent over the circuit since the counter was last zeroed. The overflow value is 4,294,967,294.

Locally initiated resets

This counter indicates the number of resets sent over the circuit since the counter was last zeroed. The overflow value is 254.

Network-initiated resets

This counter increments when a reset is sent by the PSN and is received by the local DTE over this circuit. The overflow value is 254.

Remotely initiated resets

This counter indicates the number of resets received by the local DTE that were originated by a remote DTE or DTEs over this circuit. The overflow value is 254.

E.2 Line Counters

There are five groups of line counters. Four groups are related to the Data Link layer and one group to the Network Management layer. The Data Link layer counters are divided into two groups that are device oriented and two groups that are line oriented.

E.2.1 Data Link Layer: All Devices Except DA, DMC, DMP, PCL, UNA, and QNA

Local process errors

This counter increments when the executor is responsible for a processing fault. On a multipoint line, the counter is kept as a total only for the control station, not for each tributary. Although the fault is usually caused by a programming error at the executor, there is a remote possibility that it could be caused by a line error that was not detected by the Data Link Protocol. This counter can have any of the following qualifiers: NAKs received, header format error; NAKs sent, receive overrun; receive overruns, NAK not sent. The overflow value is 254.

Remote process errors

This counter increments when an adjacent node is responsible for a processing fault. On a multipoint line, the counter is kept as a total only for the control station, not for each tributary. Although the fault is usually caused by a programming error or hardware failure at the adjacent node, there is a remote possibility that it could be caused by a line error that was not detected by the Data Link Protocol. This counter can have any of the following qualifiers: NAKs received, receive overrun; NAKs sent, header format error; selection address errors; streaming tributary. The overflow value is 254.

E.2.2 Data Link Layer: PCL Device

Attempts to become master

This counter increments when an attempt to become the master on the PCL bus fails due to other traffic on the bus. The overflow value is 254.

Device errors

This counter increments when the executor detects faulty hardware operation on its own UNIBUS. This counter can have any of the following qualifiers: interrupt timeout, receiver overflow/UNIBUS timeout, receiver overrun, transmitter overflow/UNIBUS timeout, transmitter underrun. The overflow value is 254.

Process errors

This counter increments when faulty operation is detected by the executor on one of the other nodes on the PCL bus or the executor itself. The overflow value is 254. This counter can have any of the following qualifiers:

- Flag format error; indicates a programming error at another node.
- Multiplexer address error; indicates that the executor's transmitter is not being allowed a time slice on the bus; usually caused by a hardware installation error.
- Unrecognized receiver error; indicates a hardware error during reception of the flag portion of a message.
- Unrecognized station error; indicates that the executor is missing a tributary address or that some other node is using the wrong tributary address to identify itself.

E.2.3 Network Management Layer: All Lines Except DMC

Seconds since last zeroed

This counter starts when the other line counters are zeroed. It increments by 2 every 2 seconds. This counter is zeroed when the other line counters are zeroed, so as to provide a time frame for them. The overflow value is 65,534.

E.2.4 Data Link Layer: Ethernet Lines

Blocks sent, initially deferred

This counter indicates the total number of times that a frame transmission was deferred on its first transmission attempt. It is used to measure Ethernet contention with no collisions. The overflow value is 4,294,967,294.

Blocks sent, multiple collisions

This counter indicates the total number of times that a frame was successfully transmitted on the third or later attempt, after normal collisions on previous attempts. The overflow value is 4,294,967,294.

Blocks sent, single collision

This counter indicates the total number of times that a frame was successfully transmitted on the second attempt after a normal collision on the first attempt. The overflow value is 4,294,967,294.

Bytes received

This counter increments when a data byte is received on the line. The count does not include Data Link Protocol overhead or bytes retransmitted by the Data Link layer. It can be used with the counter or data blocks received to determine the inbound traffic load on the line. The overflow value is 4,294,967,294.

Bytes sent

This counter increments when a data byte is sent on the line. The count does not include Data Link Protocol overhead or bytes retransmitted by the Data Link layer. It can be used with the counter or data blocks received to determine the outbound traffic load on the line. The overflow value is 4,294,967,294.

Collision detect check failure

This counter indicates the approximate number of times that a collision detect was not sensed after a transmission. The overflow value is 65,534.

Data blocks received

This counter increments when a data block is received on the line. The count does not include Data Link Protocol overhead. It can be used as a statistical base for evaluating the other Data Link layer counters. The overflow value is 4,294,967,294.

Data blocks sent

This counter increments when a data block is sent on the line. The count does not include Data Link Protocol overhead or blocks retransmitted by the Data Link layer. It can be used as a statistical base for evaluating the other Data Link layer counters. The overflow value is 4,294,967,294.

Data overrun

This counter indicates the total number of times that the hardware lost an incoming frame because it was unable to keep up with the data rate. The overflow value is 65,534.

Multicast blocks received

This counter indicates the total number of multicast blocks that have been successfully received. The overflow value is 4,294,967,294.

Multicast bytes received

This counter indicates the total number of multicast data bytes that have been successfully received including bytes in the Ethernet data field, but not the Ethernet data link headers. The overflow value is 4,294,967,294.

Receive failure

This counter indicates the total number of blocks received with some data error. The blocks are data frames that passed either physical or multicast address comparison. For each increment of the counter, one of the following types of failure is recorded: block check error, framing error, frame too long. The overflow value is 65,534.

Send failure

This counter indicates the total number of times that a transmit attempt failed. For each increment of the counter, one of the following types of failure is recorded: carrier check failed, excessive collisions, frame too long, open circuit, remote failure to defer, short circuit. The overflow value is 65,534.

System buffer unavailable

This counter indicates the total number of times that no system buffer was available for an incoming frame. This can be any buffer between the hardware and the user buffers which are those supplied on receive requests. The overflow value is 65,534.

Unrecognized frame destination

This counter indicates the number of times that a frame was discarded because there was no enabled portal with the protocol type or multicast address. The count includes frames received for the physical address, broadcast address, or multicast address. The overflow value is 65,534.

Bytes received

This counter increments when a data byte is received on the line. The count does not include Data Link Protocol overhead or bytes retransmitted by the Data Link layer. It can be used with the counter or data blocks received to determine the inbound traffic load on the line. The overflow value is 4,294,967,294.

Bytes sent

This counter increments when a data byte is sent on the line. The count does not include Data Link Protocol overhead or bytes retransmitted by the Data Link layer. It can be used with the counter or data blocks received to determine the outbound traffic load on the line. The overflow value is 4,294,967,294.

Data blocks received

This counter increments when a data block is received on the line. The count does not include Data Link Protocol overhead. It can be used as a statistical base for evaluating the other Data Link layer counters. The overflow value is 4,294,967,294.

Data blocks sent

This counter increments when a data block is sent on the line. The count does not include Data Link Protocol overhead or blocks retransmitted by the Data Link layer. It can be used as a statistical base for evaluating the other Data Link layer counters. The overflow value is 4,294,967,294.

Data errors inbound

This counter indicates the number of incoming data errors resulting from faults on the line between the local DTE and the DCE. The counter can have any of the following qualifiers: block too long, block check error, reject sent. The overflow value is 254.

Data errors outbound

This counter indicates the number of outgoing data errors resulting from faults on the line between the local DTE and the DCE. It can include the following qualifier: reject received. The overflow value is 254.

Local buffer errors

This counter increments when a receive not ready (RNR) packet is sent. It can include the following qualifier: RNR sent, buffer unavailable. The overflow value is 254.

Local process errors

This counter increments when a frame reject error (FRMR) is received over the line or when your system is overloaded. The counter can have any of the following qualifiers: transmit underrun; receive overrun; FRMR received, header format error. Normally, the first two qualifiers indicate that the system is overloaded, while the third indicates that the RSX–11 PSI software is malfunctioning. The overflow value is 254.

Local reply timeouts

This counter increments when the retransmit timer for that line expires. The overflow value is 254.

Remote buffer errors

This counter increments when a receive not ready (RNR) packet is received. It can include the following qualifier: RNR received, buffer unavailable. The overflow value is 254.

Remote process errors

This counter increments when an invalid next expected sequence number (N(R)) and a frame reject error (FRMR) are sent over the line. The counter can have any of the following qualifiers: invalid N(R) received; FRMR sent, header format error. Usually these errors indicate that the DCE is malfunctioning. The overflow value is 254.

Remote reply timeouts

This counter indicates the number of times that a frame with the poll bit set has been received over the line—that is, the number of errors resulting from faults on the line. The overflow value is 254.

E.3 Module Counters

There are two types of module counters: protocol module counters and server module counters. The protocol module is the X.25 protocol module. The server modules are the X.25 server and the X.29 server modules.

Bytes received

This counter increments when a byte is received by the local DTE. The overflow value is 4,294,967,294.

Bytes sent

This counter increments when a byte is sent by the local DTE. The overflow value is 4,294,967,294.

Calls received

This counter increments when an incoming call is received by the DTE. The overflow value is 65,534.

Calls sent

This counter increments when an outgoing call is sent. The overflow value is 65,534.

Data blocks received

This counter increments when a data block is received by the local DTE. The overflow value is 4,294,967,294.

Data blocks sent

This counter increments when a data block is sent by the local DTE. The overflow value is 4,294,967,294.

Fast selects received

This counter increments when a call is received with the fast select facility specified. The overflow value is 65,534.

Fast selects sent

This counter increments when a call is sent with the fast select facility specified. The overflow value is 65,534.

Locally initiated resets

This counter increments when a reset is sent by the local DTE. The overflow value is 254.

Maximum channels active

This counter indicates the number of switched virtual circuits that were active at any one time since the counters were last zeroed. These circuits are ones whose logical channel numbers appear in the channel list regardless of whether the circuits are used for incoming or outgoing calls. The overflow value is 65,534.

Maximum switched circuits active

This counter indicates the number of switched virtual circuits that were active at any one time since the counters were last zeroed. The overflow value is 65,534.

Network-initiated resets

This counter increments when a reset is sent by the PSN and is received by the local DTE. The overflow value is 254.

Received call resource errors

This counter increments when an incoming call is rejected because of insufficient resources. The overflow value is 65,534.

Remotely initiated resets

This counter indicates the number of resets received by the local DTE that were originated by a remote DTE or DTEs. The overflow value is 254.

Restarts

This counter indicates the number of times that the restart procedure was used on the DTE. The overflow value is 254.

Seconds since last zeroed

This counter starts when the other protocol module counters are zeroed. It increments by 2 every 2 seconds. This counter is zeroed when the other protocol module counters are zeroed, so as to provide a time frame for them. The overflow value is 65,534.

E.3.2 X.25/X.29 Server Modules

Incoming calls rejected, no resources

This counter indicates the number of times that the incoming call handler rejected a request to set up a virtual circuit because of insufficient resources. The overflow value is 254.

Logical links rejected, no resources

This counter increments each time that a logical link could not be established because of insufficient resources. The overflow value is 254.

Maximum circuits active

This counter indicates the number of switched virtual circuits that have been set up since the counters were last zeroed. The overflow value is 65,534.

Seconds since last zeroed

This counter starts when the other server module counters are zeroed. It increments by 2 every 2 seconds. This counter is zeroed when the other server module counters are zeroed, so as to provide a time frame for them. The overflow value is 65,534.

E.4 Node Counters

E.4.1 Network Management Layer

Seconds since last zeroed

This counter starts when the other node counters are zeroed. It increments by 2 every 2 seconds. This counter is zeroed when the other node counters are zeroed, so as to provide a time frame for them. The overflow value is 65,534.

Bytes received

This counter increments when user data bytes are received from the associated node at the logical link level. It includes only the user data from data messages and from interrupt, connect, accept, reject, disconnect, and abort functions. The overflow value is 4,294,967,294.

Bytes sent

This counter increments when user data bytes are sent to the associated node at the logical link level. It includes only the acknowledged user data from data messages and from interrupt, connect, accept, reject, disconnect, and abort functions. It does not include retransmissions. The overflow value is 4,294,967,294.

Connects received

This counter increments when a connect initiation signal is received from the associated node. The overflow value is 65,534.

Connects sent

This counter increments when a connect initiation signal is sent to the associated node. The overflow value is 65,534.

Messages received

This counter increments when a message is received from the associated node at the logical link level. The count includes both user messages and logical link protocol control messages. Furthermore, it includes internal segmentation of user messages by the Network Services layer. The overflow value is 4,294,967,294.

Messages sent

This counter increments when a message is sent to the associated node at the logical link level. The count includes both user messages and logical link protocol control messages. It also includes the retransmission of a message. The Network Services layer segments the user messages. The overflow value is 4,294,967,294.

Node maximum logical links active

This counter records the number of active links between the executor and the associated node. The overflow value is 65,534.

Received connect resource errors

This counter increments when the executor attempts to reject a connect initiation signal from the associated node due to lack of resources in the executor. This condition occurs when there are not enough maximum links allowed or when there are not enough control buffers or small buffers. The condition can also occur if there is an insufficient number of node counter blocks. The overflow value is 65,534.

Response timeouts

This counter increments when the associated node fails to respond within the required time. This situation can be caused either by messages being discarded in the network (see Section E.1.2.) or by a wide variance in the round-trip delay to the node. This condition normally indicates an overload condition in the network. This should be considered a problem if 2 percent or more of the messages sent are timed out. The overflow value is 65,534.

Total maximum logical links active

This counter records the number of active logical links between the executor and all nodes including itself. This counter is included in counters for the executor node only. The overflow value is 65,534.

Total received connect resource errors

This counter increments when the executor attempts to reject an initiation signal from any node due to the lack of resources in the executor. This counter is included in counters for the executor node. The condition is caused by the maximum logical link parameter being set too low or by the number of buffers parameter set too low for control buffers or small buffers. This condition can also occur if there is an insufficient number of node counter blocks. The overflow value is 65,534.

Aged packet loss

This counter increments when a packet is discarded due to visiting too many nodes. The count is the total of all such discards by the executor node. Only routing nodes keep this counter. This counter is incremented each time the aged packet loss event occurs. The overflow value is 254.

Node out of range packet loss

This counter increments when a packet is discarded because the destination node address was greater than the maximum address defined for the executor. The count is the total of all such discards by the executor node. Only routing nodes keep this counter. This counter is incremented each time the node out of range packet loss event occurs. The overflow value is 254.

Node unreachable packet loss

This counter increments when a packet is discarded because its destination node was unreachable. The count is the total of all such discards by the executor node. Only routing nodes keep this counter. The counter is incremented each time the node unreachable packet loss event occurs. The overflow value is 65,534.

Oversized packet loss

This counter increments when a packet is discarded because it was larger than the circuit buffer size. The circuit buffer size was previously established between the executor node and the adjacent node. Only routing nodes keep this counter. The counter is incremented each time the oversized packet loss event occurs. The overflow value is 254.

Packet format error

This counter increments when a packet is discarded because of invalid packet control information. The count is the total of all such discards by the executor node. This counter is incremented each time the packet format error event occurs. The overflow value is 254.

Partial routing update loss

This counter increments when part of a routing update is lost because it contained a reachable node address that exceeded the maximum address defined for the executor node. The count is the total of all such occurrences at the executor node. Only routing nodes keep this counter. This counter is incremented each time the partial routing update loss event occurs. The overflow value is 254.

Maximum logical links active

This counter records the maximum number of logical links that are active and increments each time the current value is exceeded.

Verification reject

This counter increments when the executor rejects a verification request from an adjacent node during routing initialization. The count is the total of all such occurrences at the executor node. This counter is incremented each time the verification reject event occurs. The overflow value is 254.

E.5 System Counters

There are five system counters: four buffer allocation failure counters and one timing counter. Any of the following buffer allocation failures can be resolved using CFE. Use the CFE LIST SYSTEM CHARACTERISTICS command to determine the present setting of the buffer in question. Then, use the CFE DEFINE SYSTEM command to increase the number of those buffers. After the number of buffers is increased, reload DECnet.

Control buffer allocation failed

This counter increments when a control buffer allocation fails. This condition is caused by not having enough control buffers to service the work load on the Communications Executive and its processes. The overflow value is 65,534.

Large buffer allocation failed

This counter increments when a large buffer allocation fails. This condition is caused by not having enough large buffers available to service the work load on the Communications Executive and its processes. It can also be a symptom of the minimum receive buffer level reserving too many large buffers. The overflow value is 65,534.

Receive buffer allocation failed

This counter increments when a receive buffer allocation fails. This condition is caused by not having enough receive buffers reserved from the large buffers to service the work load on the Communications Executive and its processes. The overflow value is 65,534.

Seconds since last zeroed

This counter starts when the other system counters are zeroed. It increments by 2 every 2 seconds. This counter is zeroed when the other system counters are zeroed, so as to provide a time frame for them. The overflow value is 65,534.

Small buffer allocation failed

This counter increments when a small buffer allocation fails. This condition is caused when there are not enough small buffers to service the work load on the Communications Executive and its processes. The overflow value is 65,534.

F

Network Parameter and Counter Type Numbers

When you execute an NCP command remotely at an RSX node from a non-RSX node, your terminal might display a parameter or counter type number in place of the usual text in a SHOW display or in an error message. For example, either of the following two equivalent displays could be generated in response to the same SHOW command.

```
NCP> TELL BOSTON SHOW LINE UNA-0 CHARACTERISTICS

Line characteristics as of 21-SEP-86 14:00:09

Line = UNA-0

   Controller = Normal
   Protocol = Ethernet
   Hardware address = AA-00-03-00-01-13
   Controller CSR = 174510, Vector = 120, Priority = 5

Line characteristics as of 21-SEP-86 14:00:09

Line = UNA-0

   Controller = Normal
   Protocol = Ethernet
   Hardware address = AA-00-03-00-01-13
   Parameter  2310 = 174510
   Parameter  2312 = 120
   Parameter  2313 = 5
```

Similarly, on a SHOW COUNTERS command, the counters could be represented by a counter type number instead of the actual counter text. This appendix lists all parameter and counter type numbers recognized by DECnet-RSX network management software. Counter and parameter type numbers

from 2300 to 2499 are RSX system specific. Chapter 2 provides detailed information on specific network management parameters. Network counters are described in detail in Appendix E. The DNA Network Management Functional Specification describes all standard type numbers for all parameters and counters.

This appendix also lists the event classes and types supported for the DECnet-RSX logging component. Appendix D provides detailed descriptions of the corresponding event messages.

F.1 Alias Parameters (RSX System Specific)

Number	Keywords
100	SCOPE
110	DESTINATION

F.2 Circuit Parameters and Counters

F.2.1 Circuit Parameters

Number	Keywords
0	STATE
1	*substate*
110	COUNTER TIMER
400	LOOPBACK NAME
800	ADJACENT NODE
810	BLOCK SIZE
900	LEVEL 1 COST

Number	Keywords
901	MAXIMUM ROUTERS
902	ROUTER PRIORITY
904	LEVEL 2 COST
906	HELLO TIMER
907	LISTEN TIMER
920	MAXIMUM RECALLS
921	RECALL TIMER
1100	OWNER
1111	USAGE
1112	TYPE
1120	DTE
1121	CHANNEL
1122	MAXIMUM DATA
1123	MAXIMUM WINDOW
1140	TRIBUTARY
2320	MULTIPOINT ACTIVE (RSX specific)

F.2.2 Circuit Counters

Some counters include qualifiers identified by bit numbers in a bit mask. These qualifiers are included in the following lists, indented under the counter to which they belong.

The following counters are kept for all circuits:

Number	Standard Text
0	Seconds since last zeroed
800	Terminating packets received
801	Originating packets sent
805	Corruption loss
810	Transit packets received
811	Transit packets sent
812	Transit congestion loss
820	Circuit down
821	Initialization failure
822	Adjacency Down
900	Peak Adjacency

The following counters are kept for DDCMP circuits:

Number	Standard text
1000	Bytes received
1001	Bytes sent
1010	Data blocks received
1011	Data blocks sent
1020	Data errors inbound
	1 NAKs sent, data field block check error
	2 NAKs sent, REP response
1021	Data errors outbound
	0 NAKs received, header block check error
	1 NAKs received, data field block check error
	2 NAKs received, REP response

Number	Standard text
1030	Remote reply timeouts
1031	Local reply timeouts
1040	Remote buffer errors
	0 NAKs received, buffer unavailable
	1 NAKs received, buffer too small
1041	Local buffer errors
	0 NAKs sent, buffer unavailable
	1 NAKs sent, buffer too small
1050	Selection intervals elapsed
1051	Selection timeouts
	0 No reply to select

The following counters are kept for Ethernet circuits:

Number	Standard text
1000	Bytes received
1001	Bytes sent
1010	Data blocks received
1011	Data blocks sent
1065	User buffer unavailable

The following counters are kept for permanent X.25 circuits:

Number	Standard text
1000	Bytes received
1001	Bytes sent
1010	Data blocks received
1011	Data blocks sent
1240	Locally initiated resets
1241	Remotely initiated resets
1242	Network-initiated resets

F.3 Line Parameters and Counters

F.3.1 Line Parameters

Number	Keywords	
0	STATE	
1	*substate*	
100	SERVICE	
110	COUNTER TIMER	
1111	DUPLEX	
1112	PROTOCOL	
1151	DEAD TIMER	
1152	DELAY TIMER	
2300	OWNER	(RSX specific)
2310	CONTROLLER CSR	(RSX specific)
2311	UNIT CSR	(RSX specific)
2312	VECTOR	(RSX specific)

Number	Keywords	
2313	PRIORITY	(RSX specific)
2321	MULTIPOINT DEAD	(RSX specific)
2330	LOCATION	(RSX specific)

F.3.2 Line Counters

Some counters include qualifiers identified by bit numbers in a bit mask. These qualifiers are included in the following lists, indented under the counter to which they belong.

The following counters are kept for point-to-point and tributary DDCMP lines:

Number	Standard text
0	Seconds since last zeroed
1100	Remote process errors
	0 NAKs received, receive overrun
	1 NAKs sent, header format error
	2 Selection address errors
	3 Streaming tributary
1101	Local process errors
	0 NAKs sent, receive overrun
	1 Receive overruns, NAK not sent
	3 NAKs received, header format error

DDCMP software does not support the following counters as specified by the network architecture:

- NAKs sent, header block check error: bit 0 for data errors inbound.

  This is counted as a data error, under bit 1, due to information not returned by driver.

- NAKs sent, buffer unavailable: bit 0 for local buffer errors.

  This is not sensed by driver.

- Incomplete reply to select: bit 1 for selection timeouts.

  This is counted as a no reply, under bit 0, due to information not sensed by driver.

- Transmit underruns: bit 2 for local process errors.

  The driver returns a device timeout error, which is then recorded as a no reply to select, bit 0 for selection timeouts, if the line is multipoint or half duplex. Otherwise it is not counted.

The following counters are kept for the PCL device:

Number	Standard text
0	Seconds since last zeroed
2410	Attempts to become master
2411	Process errors
	0 Unrecognized receiver error
	1 Unrecognized station error
	2 Flag format error
	3 Multiplexer address error
2412	Device errors
	0 Transmitter underrun
	1 Transmitter overflow/UNIBUS timeout
	2 Receiver overrun
	3 Receiver overflow/UNIBUS timeout
	4 Interrupt timeout

The following counters are kept for Ethernet lines (UNA and QNA):

Number	Standard text
0	Seconds since last zeroed
1000	Bytes received
1001	Bytes sent
1002	Multicast bytes received
1010	Data blocks received
1011	Data blocks sent
1012	Multicast blocks received
1013	Blocks sent, initially deferred
1014	Blocks sent, single collision
1015	Blocks sent, multiple collisions
1060	Send failure
	0 Excessive collisions
	1 Carrier check failed
	2 Short circuit
	3 Open circuit
	4 Frame too long
	5 Remote failure to defer
1061	Collision detect check failure
1062	Receive failure
	0 Block check error
	1 Framing error
	2 Frame too long

(continued on next page)

Number	Standard text
1063	Unrecognized frame destination
1064	Data overrun
1065	System buffer unavailable

The following counters are kept for LAPB lines:

Number	Standard text
0	Seconds since last zeroed
1000	Bytes received
1001	Bytes sent
1010	Data blocks received
1011	Data blocks sent
1020	Data errors inbound
	3 Block too long
	4 Block check error
	5 REJ sent
1021	Data errors outbound
	3 REJ received
1030	Remote reply timeouts
1031	Local reply timeouts
1040	Remote buffer errors
	2 RNR received, buffer unavailable
1041	Local buffer errors
	2 RNR sent, buffer unavailable
1100	Remote process errors
	4 Invalid N(R) received
	5 FRMR sent, header format error

Number	Standard text
1101	Local process errors
	2 Transmit underrun
	4 Receive overrun
	5 FRMR received, header format error

F.4 Logging Parameters and Events

F.4.1 Logging Parameters

Number	Keywords
0	STATE
100	NAME
200	SINK NODE
201	EVENTS

F.4.2 Logging Events

Most of the following events occur on both routing and nonrouting nodes. Events that occur only on routing nodes are flagged with an asterisk. See Appendix D for descriptions of these events. The entity associated with each event is shown next to the event number (- = none, A = area, C = circuit, L = line, M = module, and N = node).

Number	Entity	Standard text
0.0	-	Event records lost
0.1	N	Automatic node counters
0.2	L	Automatic line counters
0.3	C	Automatic line service
0.4	L	Line counters zeroed

(continued on next page)

Number	Entity	Standard text
0.5	N	Node counters zeroed
0.6	C	Passive loopback
0.7	C	Aborted service request
0.8	All	Automatic counters
0.9	All	Counters zeroed
2.0	-	Local node state change
2.1	-	Access control failure
3.0	-	Invalid message
3.1	-	Invalid flow control
3.2	N	Node database reused
4.0 *	-	Aged packet loss
4.1 *	C	Node unreachable packet loss
4.2 *	C	Node out of range packet loss
4.3	C	Oversized packet loss
4.4	C	Packet format error
4.5 *	C	Partial routing update loss
4.6	C	Verification reject
4.7	C	Circuit down, circuit fault
4.8	C	Circuit down, software fault
4.9	C	Circuit down, operator fault
4.10	C	Circuit up
4.11	C	Circuit initialization failure, circuit fault
4.12	C	Circuit initialization failure, software fault
4.13	C	Circuit initialization failure, operator fault
4.14	N	Node reachability change
4.15	C	Adjacency up

Number	Entity	Standard text
4.16	C	Adjacency rejected
4.17	A	Area reachability change
4.18	C	Adjacency down
4.19	C	Adjacency down - operator initiated
5.0	C	Locally initiated state change
5.1	C	Remotely initiated state change
5.2	C	Protocol restart received in maintenance mode
5.3	C	Send error threshold
5.4	C	Receive error threshold
5.5	C	Select error threshold
5.6	C	Block header format error
5.7	C	Selection address error
5.8	C	Streaming tributary
5.9	C	Local buffer too small
5.10	M	Restart
5.11	M	State change
5.12	M	Retransmit maximum exceeded
5.13	L	Initialization failure
5.14	L	Send failed
5.15	L,C	Receive failed
6.0	L	Data set ready transmission
6.1	L	Ring indicator transmission

(continued on next page)

Number	Entity	Standard text
6.2	L	Unexpected carrier transmission
6.3	L	Memory access error
6.4	L	Communications interface error
6.5	L	Performance error
64.1 *	-	Routing database corrupt
64.2 *	-	Routing database restored
68.14	-	Normal usage terminated
93.0	-	State change
94.0	-	DCE detected packet error

F.5 Node Parameters and Counters

F.5.1 Node Parameters

Number	Keywords
0	STATE
100	IDENTIFICATION
101	MANAGEMENT VERSION
114	HARDWARE ADDRESS
115	SERVICE NODE VERSION
120	LOAD FILE
121	SECONDARY LOADER
122	TERTIARY LOADER
123	DIAGNOSTIC FILE
125	SOFTWARE TYPE
130	DUMP FILE

Number	Keywords
135	DUMP ADDRESS
136	DUMP COUNT
140	HOST
141	HOST
150	LOOP COUNT
151	LOOP LENGTH
152	LOOP WITH
154	LOOP HELP
500	NAME
501	CIRCUIT
502	ADDRESS
510	INCOMING TIMER
511	OUTGOING TIMER
522	INCOMING PROXY (11M–PLUS only)
523	OUTGOING PROXY (11M–PLUS only)
600	ACTIVE LINKS
601	DELAY
700	NSP VERSION
710	MAXIMUM LINKS
722	INACTIVITY TIMER
723	RETRANSMIT FACTOR
810	TYPE
820	COST
821	HOPS
822	CIRCUIT
830	NEXT NODE

(continued on next page)

Number	Keywords
900	ROUTING VERSION
901	TYPE
910	ROUTING TIMER
911	SUBADDRESSES
912	BROADCAST ROUTING TIMER
920	MAXIMUM ADDRESS
921	MAXIMUM CIRCUITS
922	MAXIMUM COST
923	MAXIMUM HOPS
924	MAXIMUM VISITS
926	MAXIMUM BROADCAST NONROUTERS (11M–PLUS only)
927	MAXIMUM BROADCAST ROUTERS
932	SEGMENT BUFFER SIZE
933	MAXIMUM PATH SPLITS (11M–PLUS routers only)
2300	RECEIVE PASSWORD (RSX specific)
2301	TRANSMIT PASSWORD (RSX specific)
2310	VERIFICATION STATE (RSX specific)

F.5.2 Node Counters

Most node counters are kept on both routing and nonrouting nodes. Counters that are kept only on routing nodes are flagged with an asterisk.

Number	Standard text
0	Seconds since last zeroed
600	Bytes received
601	Bytes sent
610	Messages received
611	Messages sent
620	Connects received
621	Connects sent
630	Response timeouts
650	Buffer unavailable
700	Total maximum logical links active
710	Received connect resource errors
900 *	Aged packet loss
901 *	Node unreachable packet loss
902 *	Node out of range packet loss
903 *	Oversized packet loss
910	Packet format error
920 *	Partial routing update loss
930	Verification reject
2300	Node maximum logical links active
2310	Total received connect resource errors

F.6 Object Parameters (RSX System Specific)

Number	Keywords
500	NAME
510	COPIES
511	USER
520	VERIFICATION

F.7 Process Parameters (RSX System Specific)

Number	Keywords
0	STATE
10	LOCATION
20	MAXIMUM LINES
21	MAXIMUM CONTROLLERS
30	PARTITION

F.8 System Parameters and Counters (RSX System Specific)

F.8.1 System Parameters

Number	Keywords
10	ACTIVE CONTROL BUFFERS
20	ACTIVE SMALL BUFFERS
30	ACTIVE LARGE BUFFERS
110	MAXIMUM CONTROL BUFFERS
120	MAXIMUM SMALL BUFFERS
130	MAXIMUM LARGE BUFFERS
131	LARGE BUFFER SIZE
140	MINIMUM RECEIVE BUFFERS

F.8.2 System Counters

Number	Standard text
0	Seconds since last zeroed
10	Control buffer allocation failed
20	Small buffer allocation failed
30	Large buffer allocation failed
40	Receive buffer allocation failed

F.9 X.25 Access Module Parameters

Number	Keywords
2310	DESTINATION
2320	NUMBER
2330	SCOPE

F.10 X.25 Protocol Module Parameters and Counters

F.10.1 X.25 Protocol Module Parameters

Number	Keywords
0	STATE
100	COUNTER TIMER
1100	DTE
1101	GROUP
1110	NETWORK
1120	LINE
1140	DEFAULT DATA
1141	DEFAULT WINDOW
1150	MAXIMUM DATA
1151	MAXIMUM WINDOW

Number	Keywords
1152	MAXIMUM CLEARS
1153	MAXIMUM RESETS
1154	MAXIMUM RESTARTS
1160	CALL TIMER
1161	CLEAR TIMER
1162	RESET TIMER
1163	RESTART TIMER
1170	DTE (qualified by GROUP)
1171	NUMBER
1172	TYPE

F.10.2 X.25 Protocol Module Counters

Number	Standard text
0	Seconds since last zeroed
1000	Bytes received
1001	Bytes sent
1010	Data blocks received
1011	Data blocks sent
1200	Calls received
1201	Calls sent
1210	Fast selects received
1211	Fast selects sent

(continued on next page)

Number	Standard text
1220	Maximum switched circuits active
1221	Maximum channels active
1230	Received call resource errors
1240	Locally initiated resets
1241	Remotely initiated resets
1242	Network-initiated resets
1250	Restarts

F.11 X.25/X.29 Server Module Parameters and Counters

F.11.1 X.25/X.29 Server Module Parameters

Number	Keywords
0	STATE
100	COUNTER TIMER
300	DESTINATION
310	MAXIMUM CIRCUITS
340	OBJECT
350	PRIORITY
351	CALL MASK
352	CALL VALUE
353	GROUP
354	NUMBER
355	SUBADDRESSES

F.11.2 X.25/X.29 Server Module Counters

Number	Standard text
0	Seconds since last zeroed
200	Maximum circuits active
210	Incoming calls rejected, no resources
211	Logical links rejected, no resources

G

RSX–11 PSI Component States and State Transitions

This appendix contains the state and state transition for the RSX–11 PSI components. The appendix consists of the following eight tables:

- Table G-1: RSX–11 PSI Circuit States and Substates
- Table G-2: RSX–11 PSI Circuit State Transitions
- Table G-3: RSX–11 PSI Line States and Substates
- Table G-4: RSX–11 PSI Line State Transitions
- Table G-5: RSX–11 PSI DTE States and Substates
- Table G-6: RSX–11 PSI DTE State Transitions
- Table G-7: RSX–11 PSI Server Module States
- Table G-8: RSX–11 PSI Server Module State Transitions

Table G–1: RSX–11 PSI Circuit States and Substates

State	Substate	Meaning
ON	RUNNING	The circuit is in normal use.
	SYNCHRONIZING	The circuit is being reset, restarted, or an error has occurred.

Table G–2: RSX–11 PSI Circuit State Transitions

Old State-Substate	New State-Substate	Cause of Change
ON-RUNNING	ON-SYNCHRONIZING	A reset or restart has been received, an error was detected, or a reset was issued.
ON-SYNCHRONIZING	ON-RUNNING	The reset operation has completed and both ends of the PVC have agreed to communicate.

Table G–3: RSX–11 PSI Line States and Substates

State	Substate	Meaning
OFF	none	The line is not usable.
ON	RUNNING	The line is in normal use.
	SYNCHRONIZING	The line is being initialized and set up.
SERVICE	IDLE	The line is not in use for anything, but is reserved for loopback testing.
	LOOPING	The line is in use for loopback testing.

Table G–4: RSX–11 PSI Line State Transitions

Old State-Substate	New State-Substate	Cause of Change
OFF	ON-SYNCHRONIZING	NCP command SET LINE STATE ON has been issued.
	SERVICE-IDLE	NCP command SET LINE STATE SERVICE has been issued.
ON-RUNNING	ON-SYNCHRONIZING	NCP command SET LINE STATE OFF has been issued.
ON-SYNCHRONIZING	ON-RUNNING	The DCE has responded correctly and the line is set up.
	OFF	NCP command SET LINE STATE OFF has been issued or the DCE has responded correctly and the line has been disconnected.
SERVICE-IDLE	SERVICE-LOOPING	NCP command LOOP LINE has been issued.
	OFF	NCP command SET LINE STATE OFF has been issued.
SERVICE-LOOPING	SERVICE-IDLE	Loopback test has completed.
	OFF	NCP command SET LINE STATE OFF has been issued.

Table G–5: RSX–11 PSI DTE States and Substates

State	Substate	Meaning
OFF	RUNNING	Level 2 and level 3 software is operational but the DTE is not available for use.
	STARTING	Level 2 software is operational but level 3 software is not. The DTE is not available for use.
	SYNCHRONIZING	Level 2 and 3 software is not operational and the DTE is not available for use.
ON	RUNNING	The DTE is available for normal use.
	STARTING	Level 2 software is operational, level 3 software is starting up, and the DTE will be available for use.
	SYNCHRONIZING	Level 2 software is starting up and the DTE will be available for use.
SHUT	RUNNING	Levels 2 and 3 are operational, but the DTE is not to be used for any new activity. All existing circuits can complete their operation.
	STARTING	Level 2 software is operational and level 3 software is starting up. When the DTE is available for use, all existing circuits can complete their operation. No new activity is allowed.
	SYNCHRONIZING	Level 2 software is starting up. When the DTE is available for use, all existing circuits can complete their operation. No new activity is allowed.

Table G–6: RSX–11 PSI DTE State Transitions

Old State-Substate	New State-Substate	Cause of Change
OFF-RUNNING	ON-RUNNING	NCP command SET MODULE X25-PROTOCOL DTE STATE ON issued.
	OFF-STARTING	Level 3 software is resynchronizing.
	OFF-SYNCHRONIZING	Level 2 software is resynchronizing.
OFF-SYNCHRONIZING	ON-SYNCHRONIZING	NCP command SET MODULE X25-PROTOCOL DTE STATE ON issued.
	OFF-STARTING	Level 2 start-up is complete.
OFF-STARTING	ON-STARTING	NCP command SET MODULE X25-PROTOCOL DTE STATE ON issued.
	OFF-RUNNING	Level 3 start-up is complete.
	OFF-SYNCHRONIZING	Level 2 software is resynchronizing.
ON-RUNNING	OFF-STARTING	NCP command SET MODULE X25-PROTOCOL DTE STATE OFF issued.
	SHUT-RUNNING	NCP command SET MODULE X25-PROTOCOL DTE STATE SHUT issued.
	ON-STARTING	Level 3 software is resynchronizing.
	ON-SYNCHRONIZING	Level 2 software is resynchronizing.
ON-SYNCHRONIZING	OFF-SYNCHRONIZING	NCP command SET MODULE X25-PROTOCOL DTE STATE OFF issued.

(continued on next page)

Table G–6 (Cont.): RSX–11 PSI DTE State Transitions

Old State-Substate	New State-Substate	Cause of Change
	SHUT-SYNCHRONIZING	NCP command SET MODULE X25-PROTOCOL DTE STATE SHUT issued.
	ON-STARTING	Level 2 start-up is complete.
ON-STARTING	OFF-STARTING	NCP command SET MODULE X25-PROTOCOL DTE STATE OFF issued.
	SHUT-STARTING	NCP command SET MODULE X25-PROTOCOL DTE STATE SHUT issued.
	ON-RUNNING	Level 3 start-up is complete.
	ON-SYNCHRONIZING	Level 2 software is resynchronizing.
SHUT-RUNNING	OFF-STARTING	NCP command SET MODULE X25-PROTOCOL DTE STATE OFF issued.
	ON-RUNNING	NCP command SET MODULE X25-PROTOCOL DTE STATE ON issued.
	SHUT-STARTING	Level 3 software is resynchronizing.
	SHUT-SYNCHRONIZING	Level 2 software is resynchronizing.
SHUT-SYNCHRONIZING	OFF-SYNCHRONIZING	NCP command SET MODULE X25-PROTOCOL DTE STATE OFF issued.
	ON-SYNCHRONIZING	NCP command SET MODULE X25-PROTOCOL DTE STATE ON issued.
	SHUT-STARTING	Level 2 start-up is complete.

Table G–6 (Cont.): RSX–11 PSI DTE State Transitions

Old State-Substate	New State-Substate	Cause of Change
SHUT-STARTING	OFF-STARTING	NCP command SET MODULE X25-PROTOCOL DTE STATE OFF issued.
	ON-STARTING	NCP command SET MODULE X25-PROTOCOL DTE STATE ON issued.
	SHUT-RUNNING	Level 3 start-up is complete.
	SHUT-SYNCHRONIZING	Level 2 software is resynchronizing.

Table G–7: RSX–11 PSI Server Module States

State	Meaning
OFF	The module is not in use.
ON	The module is available for normal use.
SHUT	The module is to be closed down but only when all present activity has ceased.

Table G–8: RSX–11 PSI Server Module State Transitions

Old State	New State	Cause of Change
OFF	ON	NCP command SET MODULE X25-SERVER STATE ON issued.
ON	OFF	NCP command SET MODULE X25-SERVER STATE OFF issued.
	SHUT	NCP command SET MODULE X25-SERVER STATE SHUT issued.
SHUT	ON	NCP command SET MODULE X25-SERVER STATE ON issued.
	OFF	NCP command SET MODULE X25-SERVER STATE OFF issued.

Index

A

Access control information
  formats for, 2–5
  within an alias node name, 2–6
Access control verification
  to specify, 2–7
Access module
  see X.25 access module
Account number formats, 2–5
ACTIVE keyword, 3–11
Alias node name
  blank alias, 2–6
  defined, 2–6
  example of how to set, 2–6
  to define scope for, 2–6
Alias parameter type numbers, F–2
ALL keyword, 3–11
Area number
  use in node address, 2–3
Areas
  configuration guidelines, 2–15
  introduction, 1–4
  routing parameters, 2–17
ASSISTANT PHYSICAL ADDRESS parameter, 4–14

B

Bootstrap
  primary, 5–4
  ROM, 5–4
Broadcast address, 2–14
BROADCAST ROUTING TIMER parameter, 2–20
Buffer counter type numbers, F–19
Buffers
  allocating memory for, 6–2
  buffer types (table), 6–3
  control buffers (CCBs), 6–7
  counters (system counters), E–20
  large data buffers (LDBs), 6–4
  number required by nodes, 6–6
  parameters, 6–1
  see also SDBs, 6–7
  small buffers (SDBs), 6–7
  system parameter type numbers, F–19

C

CCBs
  information table, 6–3
  usage, 6–7
CCR
  introductory description, 1–10

CCR (cont'd.)
sample CCR session, 5–31
to reserve the console, 5–29
CETAB.MAC
to change using NETCFE.CMD as created by NETGEN, 6–2
to modify using CFE, 1–6
CFE
LIST command, general description, 3–10
use of to modify the permanent database, 1–6
Channel parameters, 2–50
CHARACTERISTICS keyword
defined, 3–10
Circuit
circuit ID
DECnet format of, 2–38
RSX–11 PSI format of, 2–39
to modify, 2–38
circuit ID,
DLM format of, 2–39
cost, and relationship to path cost (figure), 2–21
cost, to modify, 2–21
counters, 2–51, E–2
counter type numbers, F–3
DDCMP circuits, 2–36
defined, 2–36
DLM circuit parameters, 2–47, 2–48
DLM circuits, 2–37
Ethernet circuit parameters, 2–23
Ethernet circuits, 2–37
loopback test, 4–10
multipoint circuit, defined, 2–36
ownership, 2–44
parameters (table), 2–40
parameters, to specify, 2–39
parameter type numbers, F–2
PCL circuits, 2–36
point-to-point circuit, 2–36
PVC, defined, 2–37
Circuit (cont'd.)
RSX–11 PSI circuit parameters, 2–50
RSX–11 PSI circuits, 2–37
states and loading, 2–41
states and substates (table), 2–42
substates, 2–42
SVC, defined, 2–37
to shut down a DECnet circuit, 3–4
to turn on, 3–3
types, 2–36
Circuit level loopback test, 4–1
Command node, 5–2
Command summaries
NCP (full set), A–8
NCP for RSX–11S only, A–17
VNP, A–20
Configuration File Editor
see CFE
Control buffers
see CCBs
Controller CSR address
to modify, 2–33
Controller loopback test, 4–10, 4–17
Counters
for buffers (system counters), E–20
for circuits, 2–51, E–2
for lines, 2–35, E–7
for nodes, 2–12, E–16
for X.25 modules, 2–80
for X.25 protocol module, E–14
for X.25/X.29 server modules, E–16
general description, 2–56, E–1
summary, E–1
timers for logging counters, 2–58
to display, 2–57, 3–10
to log, 2–58
to zero, 2–58
type numbers, F–1
COUNTERS keyword
defined, 3–10
Counter type numbers

Counter type numbers (cont'd.)
for circuits, F–3
for lines, F–7
for nodes, F–17
for system buffers, F–19
for X.25 protocol module, F–21
for X.25/X.29 server modules, F–23

D

Data link mapping
see DLM
DCE
defined, 1–2
DDCMP
circuit counter type numbers, F–4
circuit ID format, 2–38
circuit owners, 2–44
circuit parameters (table), 2–40
circuits, 2–36
in a sample Phase IV configuration (figure), 1–2
line, defined, 2–27
line counters, 2–35
line counter type numbers, F–7
line devices, 2–29
line parameters (table), 2–32
multipoint circuit parameters, 2–45
DEAD TIMER parameter values, 2–46
DECnet
configurations, 1–2
example of Phase IV configuration (figure), 1–2
interface with RSX operating systems, 1–1
DECnet-RSX
command files, 1–7
databases and related utilities, 1–5
device drivers and processes (table), 2–61
shutdown, 3–4
DECnet-RSX (cont'd.)
start-up, 3–2
start-up, using NCP commands, 3–2
start-up, using the NETINS.CMD file, 3–2
start-up, using VNP commands, 3–3
steps to produce a running system (figure), 1–7
steps to produce a running system (list), 1–8
DECnet-RSX/PSI
start-up using NCP commands, 3–6
DECnet-RSX/PSI
to start up using VNP commands, 3–7
DEFINE CIRCUIT command
MAXIMUM DATA parameter, 2–49
MAXIMUM WINDOW parameter, 2–49
DELAY TIMER parameter values, 2–46
DELQA, 2–30
DELUA, 2–30
DEQNA, 2–30
Designated router, 2–16, 2–23
DEUNA, 2–30
Device drivers and processes (table), 2–61
Devices
for RSX–11 PSI lines, 2–30
for DDCMP lines, 2–29
for Ethernet lines, 2–30
for PCL lines, 2–29
line types (table), 2–27
DLL, 5–3
DLM
circuit ID format, 2–39
circuit parameters, 2–47
circuit parameters (table), 2–41
circuit re-call parameters, 2–48
circuits, 2–37
circuit usage parameters, 2–48
in a sample Phase IV configuration (figure), 1–2

DLM (data link mapping)
defined, 1–2
DLX as circuit owner, 2–44
DMC-11, 2–29
DMP-11, 2–29
DMP polling rates, 2–46
DMR-11, 2–29
DMV-11, 2–29
DMV polling rates, 2–46
Down-line loading
parameter list, 5–14
parameters for load files, 5–18
parameters for the target node, 5–16
Down-line system load
defined, 5–2
operator-initiated, 5–9
over Ethernet, 5–3
Downline system load
load requirements, 5–6
load sequence, 5–6
Down-line System Loader
see DLL
Down-line task load, 5–23
DPV11, 2–30
DTE
defined, 1–2
to shut down, 3–8
to specify remote DTE address, 2–47
Dump assistance multicast address, 5–21
Dumping unattended system memory, 5–20
DUP11-DA, 2–30

E

End node, 2–16
Error messages
loopback testing, 4–11
Ethernet
see also Ethernet addresses
broadcast routing timer, 2–20
Ethernet (cont'd.)
circuit counter type numbers, F–5
circuit ID format, 2–38
circuit parameters, 2–23
circuit parameters (table), 2–41
circuits, 2–37
designated router, 2–16
dump assistance multicast address, 5–21
example of Phase IV configuration (figure), 1–2
line, defined, 2–27
line counters, 2–35
line counter type numbers, F–9
line devices, 2–30
maximum number of routers, 2–24
routing parameters (table), 2–17
up-line memory dump, 5–21
Ethernet addresses
broadcast address, 2–14
format, 2–13
general description, 2–13
hardware address, 2–13
multicast address types, 2–14
multicast group address values, 2–14
physical address, 2–13, 2–14
physical address values, 2–14
Event logger interface, C–1
Event Logging
defined, 2–51
Event logging
see Logging, Events, and Event messages
Event messages
for Data Link layer, D–13
for End Communications layer, D–6
format, D–2
for Network Management layer, D–4
for Physical Link layer, D–17
for RSX system-specific events, D–18
for Session Control layer, D–6

Event messages (cont'd.)
for Transport layer, D–7
Events
see also Event messages and Logging
class and type, specifying, 2–53
defined, 2–51
entity type, 2–54
event classes (table), D–1
event list format, 2–54
event message format, D–2
list of logging events, F–11
logging, 2–51
source type, 2–54
to display information about events being logged, 3–10
EVF
introductory description, 1–10
EXECUTOR keyword
use of, 2–4
Executor node, 5–2
see also Executor command
see also Executor commands
defined, 2–2
ID string, 2–11
subaddresses, 2–12

G

Generating the Network
where NETGEN ends and network management begins, 1–7

H

Hardware address
defined, 2–13
HARDWARE ADDRESS parameter, 5–17
Hardware loopback device, 4–10
HELLO TIMER parameter
to modify, 2–23
HELP parameter
LOOP CIRCUIT command, 4–17
HLD
commands, 5–27
error handling, 5–28
introductory description, 1–10
LUN fixing, 5–28
mapping table, 5–26
to create or modify a mapping table, 5–26
to format the mapping table, 5–26
HLD (Host Task Loader), 5–23
Hops
MAXIMUM HOPS parameter, 2–19
Host Task Loader
see HLD

K

KDA
introductory description, 1–10
KMS-11 microcode dump analyzer, 4–33
KMS-11
dumping microcode, 4–32
KMS11-BD, 2–30
KMS-11 microcode dump analyzer
see KDA
KMS11-PX, 2–30
KMV11-A, 2–30
KMV interface, 2–30
KMX-DUMP command, 4–32
KMX interface, 2–30
KMX/KMY/KMV
dumping microcode, 4–32
KMY interface, 2–30
KNOWN CIRCUITS keyword, 2–38
KNOWN keyword, 3–11
KNOWN LINES keyword, 2–30
KNOWN NODES keyword
defined, 2–4

L

LAPB
  line counter type numbers, F–10
Large data buffers
  see LDBs
LCP
  introductory description, 1–12
LDBs
  information table, 6–3
  number required by different device types (table), 6–5
  to determine number to allocate, 6–5
  to determine size of, 6–4
  usage, 6–4
Line
  counters, 2–35, E–7
  counter type numbers, F–7
  DDCMP devices, 2–29
  device types (table), 2–27
  Ethernet devices, 2–30
  line ID format, 2–30
  parameters (table), 2–32
  parameters, loaded vs. loading options, 2–31
  parameters, to specify, 2–31
  parameter type numbers, F–6
  PCL devices, 2–29
  RSX–11 PSI devices, 2–30
  states, to set and display, 2–34
  states and substates (table), 2–42
  to load, 2–34
  to load/turn on, 3–3
  to shut unload a DECnet line, 3–4
  types, 2–27
LIST command
  general description, 3–10
Load assistance multicast address, 5–3
Loading/turning on a line, 3–3
LOAD NODE command, 5–2, 5–12
  DIAGNOSTIC FILE parameter, 5–14
  FROM load file parameter, 5–12
LOAD NODE command (cont'd.)
  HOST parameter, 5–12
  overriding default parameters, 5–12
  SECONDARY LOADER parameter, 5–14
  SERVICE DEVICE parameter, 5–14
  SERVICE PASSWORD parameter, 5–14
  TERTIARY LOADER parameter, 5–14
LOAD VIA command, 5–12
Local loopback test, 4–8
Local node
  defined, 2–2
Local-to-local loopback test, 4–7
Local-to-remote loopback test, 4–6
Logging
  see also Events and Logging commands
  commands, summarized, 2–51
  components, 2–55
  event classes (table), D–1
  event logger interface, C–1
  event logging, coding example, 3–14
  event message format, D–2
  events, list of, F–11
  parameters (table), 2–52
  parameter type numbers, F–11
Log-in ID
  defined, 2–5
Loopback assistance, 4–14
Loopback connector, 4–10
Loopback mirror, 4–3
Loopback test
  circuit, 4–10
  circuit level, 4–1
  controller, 4–10, 4–17
  local node, 4–8
  local-to-local, 4–7
  local-to-remote, 4–6
  node level, 4–1
  software, 4–10, 4–11
  to a remote node, 4–4

Loopback test (cont'd.)
using a loop node name, 4–5
X.25 line level, 4–24
LOOP CIRCUIT command, 4–10
ASSISTANT NODE parameter, 4–14
ASSISTANT PHYSICAL ADDRESS parameter, 4–14
HELP parameter, 4–17
LOOP EXECUTOR command, 4–8
LOOP NODE command, 4–3
CIRCUIT parameter, 4–5
Loop node name, 4–5
LUN fixing, 5–28

M

Mailbox
contents returned by GND$, C–2
defined, C–1
Maintenance Operation Protocol
see MOP
Mapping table
see HLD
MAXIMUM ADDRESS parameter
to modify, 2–19
MAXIMUM COST parameter, 2–19
MAXIMUM DATA parameter
for PVC, 2–49
MAXIMUM HOPS parameter, 2–19
MAXIMUM PATH SPLITS parameter, 2–21
MAXIMUM WINDOW parameter
for PVC, 2–49
Microcode
dumping KMS-11, 4–32
dumping KMX/KMY/KMV, 4–32
Modem, 4–10
MOP (Maintenance Operation Protocol), 5–2, 5–20
MOP (Maintenance Operation Protocol) (cont'd.)
request dump service message, 5–20
memory dump data message, 5–20
mode running message, 5–20
Multicast address
dump assistance, 5–21
load assistance, 5–3
Multicast address types, 2–14
Multicast group address values, 2–14
Multipoint circuit
DDCMP parameters, 2–45
defined, 2–36
operation, 2–45
Multipoint controller polling rates, 2–46

N

NCP
command set descriptions, 1–6
command summary (full set), A–8
command summary for RSX–11S only, A–17
SHOW command
general description, 3–10
sample displays, 3–12
to send display information to a file, 3–12
use of to modify the volatile database, 1–6
use of to start up DECnet-RSX, 3–2
use of to start up DECnet-RSX/PSI, 3–6
NDA
introductory description, 1–11
NETCFE.CMD file, 1–7, 6–2
NETCFE.CMD file (figure), 6–2
NETGEN
where NETGEN ends and network management begins, 1–7
NETINS.CMD file, 1–9

NETINS.CMD file (cont'd.)
used to start up DECnet-RSX, 3–2
NETREM.CMD file, 1–7
Network
testing, 4–1
verification state, 8–31
Network buffer parameters, 6–1
Network Control Program
see NCP
Network management
command summary, A–1
component descriptions, 2–1
counter type numbers, F–1
parameter type numbers, F–1
responsibilities of network manager, 1–5
steps to produce a running system (figure), 1–7
steps to produce a running system (list), 1–8
Network management overview, 1–1
Network management tools
see CCR, EVF, HLD, KDA, NDA, NTD, QUE, TRI, LCP
Node
access control information requirements, 2–4
access control verification, 2–7
alias node names, 2–6
area routing node, 2–16
buffer requirements, 6–6
counters, E–16
counters general description, 2–12
counter type numbers, F–17
end node (nonrouting), 2–16
Ethernet address of node, 2–13
level 1 routing node, 2–16
level 2 routing node, 2–16
loop node defined, 2–4
node ID defined, 2–3
node name defined, 2–3
node number defined, 2–3
Node (cont'd.)
parameters, 2–7
parameters (table), 2–9
parameter type numbers, F–14
Phase III, 7–2
to shut down, 3–4
to start up, 3–2
types, 2–2, 2–16
Node address
defined, 2–3
for Ethernet, 2–13
MAXIMUM ADDRESS parameter to modify, 2–19
Node ID
defined, 2–3
Node level loopback test, 4–1
logical link operation, 4–2
over specific circuit, 4–2
Node name
defined, 2–3
Node number
defined, 2–3
NTD
introductory description, 1–11

O

Object
access control verification, 2–7
defined, 2–24
multicopy objects, 2–26
name format, 2–26
parameters (table), 2–25
parameter type numbers, F–18
UICs, 2–26
Object type codes
table, B–1
values, 2–24
Operator-initiated down-line load, 5–9

P

Parameter type numbers
  for alias parameters, F–2
  for circuit parameters, F–2
  for line parameters, F–6
  for logging parameters, F–11
  for node parameters, F–14
  for object parameters, F–18
  for process parameters, F–18
  for system (buffer) parameters, F–19
  for X.25 access module parameters, F–20
  for X.25 protocol module parameters, F–20
  for X.25/X.29 server module parameters, F–22
Password
  format of for access control, 2–5
Path cost
  MAXIMUM COST parameter, 2–19
  relationship to circuit cost (figure), 2–21
PCL
  circuit ID format, 2–38
  circuit parameters (table), 2–41
  circuits, 2–36
  device counter type numbers, F–8
  line, defined, 2–27
  line devices, 2–29
Permanent database
  CFE command functions, 2–1
  defined, 1–6
  to modify using CFE, 1–6
Permanent virtual circuit
  defined, 2–37
Phase IV sample configuration (figure), 1–2
Physical address
  defined, 2–13
  PHYSICAL ADDRESS parameter, 5–17
  set by the HARDWARE ADDRESS parameter, 5–17
Physical address (cont'd.)
  values, 2–14
Point-to-point circuit, 2–36
Polling
  for DMP and DMV multipoint controllers, 2–46
  general description, 2–45
  ratios, 2–45
Primary loader, 5–6
Process
  names and types (table), 2–61
  general description, 2–59
  MAXIMUM CONTROLLERS parameter, 2–63
  MAXIMUM LINES parameter, 2–63
  parameters (table), 2–60
  parameter type numbers, F–18
  states, 2–63
  to load, 2–63
Program load request
  over Ethernet, 5–3
Protocol module
  see X.25 protocol module
Proxy Access, 2–7
  introductory description, 1–12
  NVP, 8–31
Proxy Utility
  introductory description, 1–12
PSDN
  defined, 1–2
PVC
  defined, 2–37
PVCs
  channel parameter, 2–49
  DTE parameter, 2–49

Q

QNA, 2–30
  QUE introductory description, 1–11

Quoted string, 2–5, 2–11

R

Remote node
  defined, 2–2
  loopback test, 4–4
Routing
  defined, 2–15
  designated router, 2–23
  MAXIMUM ROUTERS parameter for Ethernet, 2–24
  parameters (table), 2–17
  timers, 2–20
Routing layer
  as circuit owner, 2–44
  routing function, 2–15
ROUTING TIMER parameter, 2–20
RSX–11 PSI
  see X.25 access module
  see X.25/X.29 server modules
  channel parameters, 2–50
  circuit ID format, 2–39
  circuit parameters, 2–50
  circuit parameters (table), 2–41
  circuits, 2–37
  defined, 1–2
  device drivers and processes (table), 2–61
  dumping KMS-11 microcode, 4–1, 4–32
  dumping KMX/KMY/KMV microcode, 4–1
  line, defined, 2–27
  line counters, 2–35
  line devices, 2–30
  line level loopback test, 4–1, 4–24
  line parameters, 2–34
  line parameters (table), 2–32
  logical destination names, to create, 2–79
  module counters, 2–80
RSX–11 PSI (cont'd.)
  module types, defined, 2–64
  start-up, 3–6
  test facilities, 4–1
  to load/turn on a line, 3–7
  to shut down a line or DTE, 3–8
  to shut down a module, 3–9
  tracing, 4–1
RSX–11 PSI,
  X.25 protocol module
    see X.25 protocol module
RSX–11 PSI tracing, 4–32
RSX–11S
  down-line load of system, 5–2
  NCP command summary, A–17
  NETGEN procedure, 5–25
  task load, 5–23

S

Satellite node, 5–20
Satellite Task Loader
  see SLD
Scope for specifying aliases, 2–6
SDBs
  information table, 6–3
  usage, 6–7
Secondary loader, 5–6, 5–18
Server modules
  see X.25/X.29 server modules
SERVICE DEVICE parameter, 5–17
SERVICE PASSWORD parameter, 5–17
SET CIRCUIT command
  SERVICE parameter, 5–8, 5–23
  STATE parameter, 5–23
SET NODE command
  HOST parameter, 5–18
  SERVICE CIRCUIT parameter, 5–9, 5–12
  SERVICE DEVICE parameter, 5–17
SHOW command

SHOW command (cont'd.)
general description, 3–10
sample displays, 3–12
Shutting down RSX–11 PSI, 3–8
Shutting down DECnet-RSX, 3–4
SIGNIFICANT keyword, 3–11
SLD, 5–26
SLD (Satellite Loader)
building, 5–25
SLD (Satellite Task Loader), 5–23
SLD LUN fixing, 5–28
Small data buffers
see SDBs
Software loopback test, 4–10, 4–11
Starting up DECnet-RSX, 3–2, 3–3
Starting up DECnet-RSX/PSI, 3–6
STATUS keyword
defined, 3–10
SUBADDRESSES parameter values, 2–12
SUMMARY keyword
defined, 3–10
SVC
defined, 2–37
Switched virtual circuit
see SVC
System counters, E–20
System counter type numbers, F–19
System image file
to modify using VNP, 1–7
VNP command functions, 2–1
System image file defined, 1–7
System parameter type numbers, F–19

T

Target-initiated down-line load, 5–4
Target node, 5–2
Tertiary loader, 5–6, 5–18
Testing the network, 4–1
Tracing facility
RSX–11 PSI, 4–32
TRI
introductory description, 1–11
X.25 Trace Interpreter Task, 4–32
Tributaries
dead polling rates, 2–46
polling of, 2–45
types of, 2–45
TRIGGER command, 5–2, 5–9
Trigger operation
bootstrap ROM, 5–4
primary bootstrap, 5–4
primary loader, 5–4
TRIGGER command, 5–9
Type numbers for parameters and counters, F–1

U

UICs, 2–26
UNA, 2–30
loopback test, 4–13
Unattended system
memory dump, 5–20
satellite, 5–20
UNIT CSR parameter, 2–33
Up-line memory dump
definition, 5–20
over Ethernet, 5–21
procedures, 5–20
requirements, 5–22
RSX–11S system, 5–20
Usage
DLM circuit parameters, 2–48
User ID
defined, 2–5

V

Virtual Network Processor
see VNP
VMR, 5–25

VNP
Command summary, A–20
SHOW command, general description, 3–10
use of to modify the system image file, 1–7
use of to start up DECnet-RSX, 3–3
use of to start up DECnet-RSX/PSI, 3–7
Volatile database
defined, 1–6
NCP command functions, 2–1
to modify using NCP, 1–6

W

Wildcard character
in circuit IDs, 2–38
in event lists, 2–54
in line IDs, 2–31

X

X.25
line level loopback test, 4–24
trace interpreter task, 4–32
X.25 access module
defined, 2–79
parameter type numbers, F–20
X.25 circuit counter type numbers, F–5
X.25 protocol module
counters, 2–80, E–14
counter type numbers, F–21
defined, 2–64
parameters (table), 2–64
parameter type numbers, F–20
X.25 server modules
to set the state for, 2–79
to shut down, 2–79
X.25 Trace Interpreter Task
see TRI
X.25/X.29 server modules
counters, 2–80, E–16
counter type numbers, F–23
defined, 2–72
parameter type numbers, F–22
to shut down, 3–9
XPT as circuit owner, 2–44

HOW TO ORDER ADDITIONAL DOCUMENTATION

DIRECT TELEPHONE ORDERS

In Continental USA
call 800–DIGITAL

In Canada
call 800–267–6215

In New Hampshire
Alaska or Hawaii
call 603–884–6660

In Puerto Rico
call 809–754–7575 x2012

ELECTRONIC ORDERS (U.S. ONLY)

Dial 800–DEC–DEMO with any VT100 or VT200
compatible terminal and a 1200 baud modem.
If you need assistance, call 1–800–DIGITAL.

DIRECT MAIL ORDERS (U.S. AND PUERTO RICO*)

DIGITAL EQUIPMENT CORPORATION
P.O. Box CS2008
Nashua, New Hampshire 03061

DIRECT MAIL ORDERS (Canada)

DIGITAL EQUIPMENT OF CANADA LTD.
940 Belfast Road
Ottawa, Ontario, Canada K1G 4C2
Attn: A&SG Business Manager

INTERNATIONAL

DIGITAL
EQUIPMENT CORPORATION
A&SG Business Manager
c/o Digital's local subsidiary
or approved distributor

Internal orders should be placed through the Software Services Business (SSB)
Digital Equipment Corporation, Westminster, Massachusetts 01473

*Any prepaid order from Puerto Rico must be placed
with the Local Digital Subsidiary:
809–754–7575 x2012

READER'S COMMENTS

What do you think of this manual? Your comments and suggestions will help us to improve the quality and usefulness of our publications.

Please rate this manual:

	Poor				Excellent
Accuracy	1	2	3	4	5
Readability	1	2	3	4	5
Examples	1	2	3	4	5
Organization	1	2	3	4	5
Completeness	1	2	3	4	5

Did you find errors in this manual? If so, please specify the error(s) and page number(s).

_____

_____

_____

_____

General comments:

_____

_____

_____

_____

Suggestions for improvement:

_____

_____

_____

_____

Name _____ Date _____

Title _____ Department _____

Company _____ Street _____

City _____ State/Country _____ Zip Code _____

DO NOT CUT – FOLD HERE AND TAPE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY LABEL
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

digital™

Networks and
Communications Publications
550 King Street
Littleton, MA 01460–1289

DO NOT CUT – FOLD HERE