DECnet Digital Network Architecture
Phase IV
Network Management Functional Specification
Order No. AA-X437A-TK

digital

software

DECnet Digital Network Architecture Phase IV

Network Management Functional Specification

Order No. AA-X437A-TK

December 1983

This document describes the functions, structures, protocols, algorithms, and operation of the Digital Network Architecture Network Management modules. It is a model for DECnet implementations of Network Management software. Network Management provides control and observation of DECnet network functions to users and programs.

SUPERSESSION/UPDATE INFORMATION: This is a new manual.

VERSION: 4.0.0

To order additional copies of this document, contact your local Digital Equipment Corporation Sales Office.

digital equipment corporation · maynard, massachusetts

First Printing, December 1983

The postage-prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

Distributed Systems Publications typeset this manual using DIGITAL's TMS–11 Text Management System.

MGTPEALL

CONTENTS

1	INTRODUCTION	7
1.1	Intended Audience	7
1.2	Network Management Architecture, DECnet Networks, and DNA	7
1.3	Protocols and Interfaces	7
1.4	Requirements for Implementations	7
1.5	Related Documents	8
2	FUNCTIONAL DESCRIPTION	9
2.1	Design Scope	10
2.2	Relationship to DIGITAL Network Architecture	11
2.3	Network Management Model of DNA from the User Perspective	14
2.3.1	Nodes	15
2.3.2	Areas	16
2.3.3	Logging	17
2.3.4	Circuits and Lines	17
2.3.5	Modules	18
2.4	Model of Network Management Components in the DNA Layers	19
3	NETWORK MANAGEMENT AS SEEN BY THE USER	23
3.1	Nodes	23
3.1.1	Node Parameters	25
3.1.2	Node Counters	35
3.2	Areas	36
3.3	Logging	37
3.3.1	Source Qualifiers	38
3.3.2	Logging Parameters	38
3.4	Circuits	39
3.4.1	Circuit Parameters	40
3.4.2	Circuit Counters	48
3.5	Lines	49
3.5.1	Line Parameters	49
3.5.2	Line Counters	54
3.6	Circuit and Line State and Substate Model	55
3.6.1	High Level Link User States	56
3.6.2	Data Link States	58
3.6.3	Network Management Data Link Service States	60
3.6.4	Controllable and Observable States and Substates	63
3.7	Modules	65
3.7.1	X.25 Access Module	65
3.7.2	X.25 Protocol Module	66
3.7.3	X.25 Server Module	72
3.7.4	Link Maintenance Modules	75
3.8	Events	78
3.8.1	Events Not Related to an Entity	78
3.8.2	Node Events	79
3.8.3	Circuit Events	79
3.8.4	Line Events	80
3.8.5	Module Events	81
3.9	Event Parameters	81

3.9.1 Network Management Layer 81
3.9.2 Session Control Layer 82
3.9.3 End Communication Layer 83
3.9.4 Routing Layer 84
3.9.5 Data Link Layer 85
3.9.6 Physical Link Layer 87

4 NETWORK CONTROL PROGRAM (NCP) 88
4.1 Network Control Program Functions 88
4.1.1 Changing Parameters 89
4.1.2 Gathering Information 89
4.1.3 Down-line Loading 90
4.1.4 Up-line Dumping 90
4.1.5 Triggering Bootstrap 90
4.1.6 Testing Link and Network 90
4.1.7 Zeroing Counters 91
4.2 Network Control Program Operation 91
4.2.1 Specifying the Executor 91
4.2.2 Program Invocation, Termination, and Prompting 91
4.2.3 Privileged Commands 92
4.2.4 Input Formats 92
4.2.5 Output Characteristics 94
4.2.6 Status and Error Messages 94
4.3 Network Control Program Commands 96
4.3.1 SET and DEFINE Commands 96
4.3.2 CLEAR and PURGE Commands 104
4.3.3 TRIGGER Commands 108
4.3.4 LOAD Commands 109
4.3.5 DUMP Commands 110
4.3.6 LOOP Commands 111
4.3.7 SHOW QUEUE Command 112
4.3.8 SHOW and LIST Commands 112
4.3.9 ZERO Commands 118
4.3.10 EXIT Command 119

5 NETWORK MANAGEMENT OPERATION 120
5.1 NICE Access Routines and Listener 120
5.2 Local Network Management Functions 121
5.3 Link Watcher 122
5.4 Data Link Service Functions 122
5.4.1 States and Substates 123
5.4.2 Priority Control 124
5.4.3 Link State Algorithms 125
5.4.4 Link Handling Functions 126
5.5 Event Logger 127
5.5.1 Event Logger Components 129
5.5.2 Suggested Formats for Logging Data 132
5.6 Down-line Load Operation 133
5.7 Up-line Dump Operation 138
5.8 Trigger Bootstrap Operation 139
5.9 Loop Test Operation 140
5.9.1 Node Level Testing 140
5.9.2 Data Link Testing 149
5.10 Change Parameter Operation 152
5.11 Read Information Operation 154

5.12 Zero Counters Operation 154
5.13 Loopback Mirror Operation 154
5.14 NICE Logical Link Handling 155
5.15 Algorithm for Accepting Version Numbers 157
5.16 Return Code Handling 157

6 NETWORK MANAGEMENT MESSAGES 159
6.1 NICE Function Codes 159
6.2 Message Format Notation 160
6.3 Request Down-line Load Message Format 161
6.4 Request Up-line Dump Message Format 163
6.5 Trigger Bootstrap Message Format 164
6.6 Test Message Format 165
6.7 Change Parameter Message Format 166
6.8 Read Information Message Format 167
6.9 Zero Counters Message Format 168
6.10 NICE System Specific Message Format 168
6.11 NICE Response Message Format 169
6.12 NICE Connect Initiate and Connect Accept Data Formats 170
6.13 Event Message Binary Data Format 170
6.14 Logical Loopback Message Formats 172
6.14.1 Connect Accept Data Format 172
6.14.2 Command Message Format 172
6.14.3 Response Message Format 172

7 PARAMETER AND COUNTER BINARY FORMATS AND VALUES . 173
7.1 Introduction to Binary Format Descriptions . . . 173
7.1.1 Type Numbers 173
7.1.2 Entity Parameter Identifier Format 173
7.1.3 String Identifier Format 174
7.1.4 Node Identifier Formats 174
7.1.5 Area Identifier Format 175
7.1.6 Object Format for Entity Types 175
7.1.7 Numeric Range 176
7.1.8 Parameter Display Format and Descriptive Encoding Notation 176
7.1.9 NICE Returns 176
7.1.10 Information Types 179
7.1.11 Applicability Restrictions 179
7.1.12 Setability Restrictions 179
7.2 Circuit Parameters 180
7.3 Circuit Counters 183
7.4 Line Parameters 185
7.5 Line Counters 187
7.6 Logging Parameters 189
7.7 Module Parameters 191
7.7.1 Console Module Parameters 191
7.7.2 Loader Module Parameters 191
7.7.3 Looper Module Parameters 192
7.7.4 Configurator Module Parameters 192
7.7.5 X.25 Access Module Parameters 194
7.7.6 X.25 Protocol Module Parameters 194
7.7.7 X.25 Server Module Parameters 196
7.8 Module Counters 197

7.8.1 X.25 Protocol Module Counters 197
7.8.2 X.25 Server Module Counters 197
7.9 Node Parameters 197
7.10 Node Counters 201
7.11 Area Parameters 202
7.12 Event Definitions 202
7.13 Event Parameters 205

APPENDIX A VERSION COMPATIBILITY

A.1 Versions 2.0 and 3.0 . . . . A-1
A.1.1 Module Entity A-2
A.1.2 Node Entity A-2
A.1.3 Logging Entity A-2
A.1.4 Circuit and Line Entities A-2
A.1.5 Event Logging A-4
A.2 Versions 3.0 and 4.0 . . . . A-4

APPENDIX B MINIMUM SUBSET

APPENDIX C STATE MAPPING TABLES

APPENDIX D X.25 NATIVE ONLY SUBSET

APPENDIX E MEMORY IMAGE FORMATS AND FILE CONTENTS

APPENDIX F NICE RETURN CODES WITH EXPLANATIONS

APPENDIX G NCP COMMAND STATUS AND ERROR MESSAGES

APPENDIX H JULIAN HALF-DAY ALGORITHMS

APPENDIX I DMC DEVICE COUNTERS

APPENDIX J GLOSSARY

1 INTRODUCTION

1.1 Intended Audience

This document is written primarily for those who implement Network Management on DECnet systems. However, it may also be of interest to anyone who wants to know the details of the Network Management structure. Knowledge of communications software technology, DECnet, and X.25 is prerequisite to understanding this document.

Sections 1-4 describe Network Management mainly from the user perspective. Sections 5-7 describe Network Management internals.

1.2 Network Management Architecture, DECnet Networks, and DNA

This document describes the structure, functions, operation, and protocols of Network Management. Network Management models software that enables operators and programs to plan, control, and maintain the operation of centralized or distributed DECnet networks. Networks consist of software modules, data bases, and hardware components that connect computing systems for resource sharing, distributed computation, or remote system communication. DECnet networks connect DIGITAL computing systems together, and also connect to public data networks with X.25 circuits.

Network Management is part of the DIGITAL Network Architecture (DNA). DNA is the model on which DECnet network software implementations are based.

1.3 Protocols and Interfaces

DNA is a layered structure. Modules in each layer perform distinct functions. Modules within the same layer (either in the same or different nodes) communicate using specific protocols. The protocols specified in this document are the Network Information and Control Exchange (NICE) protocol, the Loopback Mirror protocol, and the Event Receiver protocol.

Modules in different layers interface using subroutine calls or a similar system-dependent method. This document describes Network Management's interface to each layer by describing elements in each layer that Network Management controls or examines.

1.4 Requirements for Implementations

This document describes user commands that can be standardized across different DECnet implementations. An implementation may use only a subset of the commands described herein. (Appendix B describes the minimum subset of Network Management functions required for certification.) Moreover, commands and functions specific to one particular operating system are not described herein.

This document uses both algorithms and English descriptions to explain the Network Management functions. An implementation is not required to follow these algorithms exactly, as long as the functions operate as described.

1.5 Related Documents

This is one of a series of functional specifications for the DIGITAL Network Architecture, Phase IV. The other current DNA functional specifications are:

DNA Data Access Protocol (DAP) Functional Specification, Version 5.6.0, Order No. AA-K177A-TK

DNA Digital Data Communications Message Protocol (DDCMP) Functional Specification, Version 4.1.0, Order No. AA-K175A-TK

DNA Ethernet Data Link Functional Specification, Version 1.0.0, Order No. AA-Y298A-TK

DNA Ethernet Node Product Architecture Specification, Version 1.0.0, Order No. AA-X440A-TK

DNA Maintenance Operations Functional Specification, Version 3.0.0, Order No. AA-X436A-TK

DNA Network Services Protocol Functional Specification, Version 4.0.0, Order No. AA-X439A-TK

DNA Routing Layer Functional Specification, Version 2.0.0, Order No. AA-X435A-TK

DNA Session Control Functional Specification, Version 1.0.0, Order No. AA-K182A-TK

The Ethernet - A Local Area Network- Data Link Layer and Physical Layer Specifications, Version 2.0, (Digital, Intel, and Xerox), Order No. AA-K759B-TK

The DECnet DIGITAL Network Architecture (Phase IV) General Description (Order No. AA-N149A-TC) provides an overview of the network architecture and an introduction to each of the functional specifications.

2 FUNCTIONAL DESCRIPTION

Network Management enables operators and programs to control and monitor network operation. Network Management helps the manager of a network to plan its evolution. Network Management also facilitates detection, isolation, and resolution of conditions that impede effective network use.

Network Management provides user commands and capability to user programs for performing the following control functions:

1. Loading remote systems. A system in one node can down-line load a system in another node in the same network.

2. Configuring resources. A system manager can change the network configuration and modify message traffic patterns.

3. Setting parameters. Circuit, line, module, node, and logging parameters (for example, node names) can be set and changed.

4. Initiating and terminating network functions. A system manager or operator can turn the network on or off and perform loopback tests and other functions.

Network Management also enables the user to monitor network functions, configurations, and states, as follows:

1. Dumping remote systems. A system in one node can up-line dump a system to another node in the same network.

2. Examining configuration status. Information about lines and nodes can be obtained. For example, an operator can display the states of lines and nodes or the names of adjacent nodes.

3. Examining parameters. Parameters (for example, timer settings, line type, or node names) can be read.

4. Examining the status of network operations. An operator can monitor network operations. For example, the operator can find out what operations are in progress and whether any have failed.

5. Examining performance variables. A system manager can examine the contents of counters in lower DNA layers to measure network performance. In addition, Network Management's Event Logger provides automatic logging of significant network events.

Besides controlling and monitoring the day-to-day operation of the network, the functions listed above work to collect information for future planning. These functions furnish basic operations (primitives) for detecting failures, isolating problems, and repairing and restoring a network.

2.1 Design Scope

Network Management functions satisfy the following design requirements:

1. Common interfaces. Common interfaces are provided to operators and programs, regardless of network topology or configuration, as much as possible without impacting the quality of existing products. There is a compromise between the compatibility of network commands across heterogeneous systems and the compatibility within a system between network and other local system commands.

2. Subsetability. Nodes are able to support a subset of Network Management components or functions.

3. Ease of use. Invoking and understanding Network Management functions are easy for the operator or user programmer.

4. Network efficiency. Network Management is both processing and memory efficient. It is line efficient where this does not conflict with other goals.

5. Extensibility. There is accommodation for future, additional management functions, leaving earlier functions as a compatible subset. This specification serves as a basis for building more sophisticated network management programs.

6. Heterogeneity. Network Management operates across a mixture of network node types, communication lines, topologies, and among different versions of Network Management software.

7. Robustness. The effects of errors such as operator input errors, protocol errors, and hardware errors are minimized.

8. Security. Network Management supports the existing security mechanisms in the DIGITAL Network Architecture (for example, the access control mechanism of the Session Control Layer).

9. Simplicity. Complex algorithms and data bases are avoided. Functions provided elsewhere in the architecture are not duplicated.

10. Support of diverse management policies. Network Management covers a range between completely centralized and fully distributed management.

11. Integrated abstractions. Diverse low level policies, such as different Data Link protocols, are combined where possible into consistent higher level abstractions.

The following are not within the scope of this version of Network Management:

1. Accounting. This specification does not provide for the recording of usage data that would be used to keep track of individual accounts for purposes of reporting on or charging users.

2. Automation. This specification does not provide for automatic execution of complex algorithms that handle network repair or reconfiguration. More automation can be expected in future revisions of this specification.

3. Protection against malicious use. There is no foolproof protection against malicious use or gross errors by operators or programs.

4. Upward compatibility of user interfaces. The interfaces to the User Layer are not necessarily frozen with this version. Observable data may change with the next version. Version 4.0 is compatible with Version 3.0 except for the changes necessary to distinguish network areas, while Version 3.0 is compatible with Version 2.0 except for those changes necessitated by the integration of X.25 Network Management functions, the DMP device, and multipoint software functions. (See Appendix A.) Compatibility with versions before Version 2.0 is not supported.

2.2 Relationship to DIGITAL Network Architecture

DIGITAL Network Architecture (DNA), the model upon which DECnet implementations are based, outlines several functional layers, each with its own specific software components, protocols, and interfaces to adjacent layers. Network Management software components reside in the three highest layers.

The general design of DNA is as follows in order from the highest to the lowest layer:

1. The User Layer. The User Layer is the highest layer. It supports user services and programs. The Network Control Program (NCP) resides in this layer.

2. The Network Management Layer. The Network Management Layer is the only one that has direct access to each lower layer for control purposes. Software components in this layer provide user control over, and access to, network parameters and counters as well as up-line dumping, down-line loading, and testing functions.

3. The Network Application Layer. Software components in the Network Application Layer support I/O device and file access functions. The Network Management software component in this layer is the Loopback Mirror, providing logical link loopback testing.

4. The Session Control Layer. The Session Control Layer manages the system-dependent aspects of logical link communication.

5. The End Communication Layer. The End Communication Layer controls the creation, maintenance, and destruction of logical links, using the Network Services Protocol.

6. The Routing Layer. Software components in the Routing Layer route messages between source and destination nodes.

7. The Data Link Layer. The Data Link Layer manages the communications over a physical link, using a data link protocol, for example, the Digital Data Communications Message Protocol (DDCMP) or the X.25 Protocol.

8. The Physical Link Layer. The Physical Link Layer provides the hardware interfaces (such as EIA RS-232-C or CCITT V.24) to specific system devices.

Figure 1 shows the relationship of the Network Management Layer to the other DNA layers.

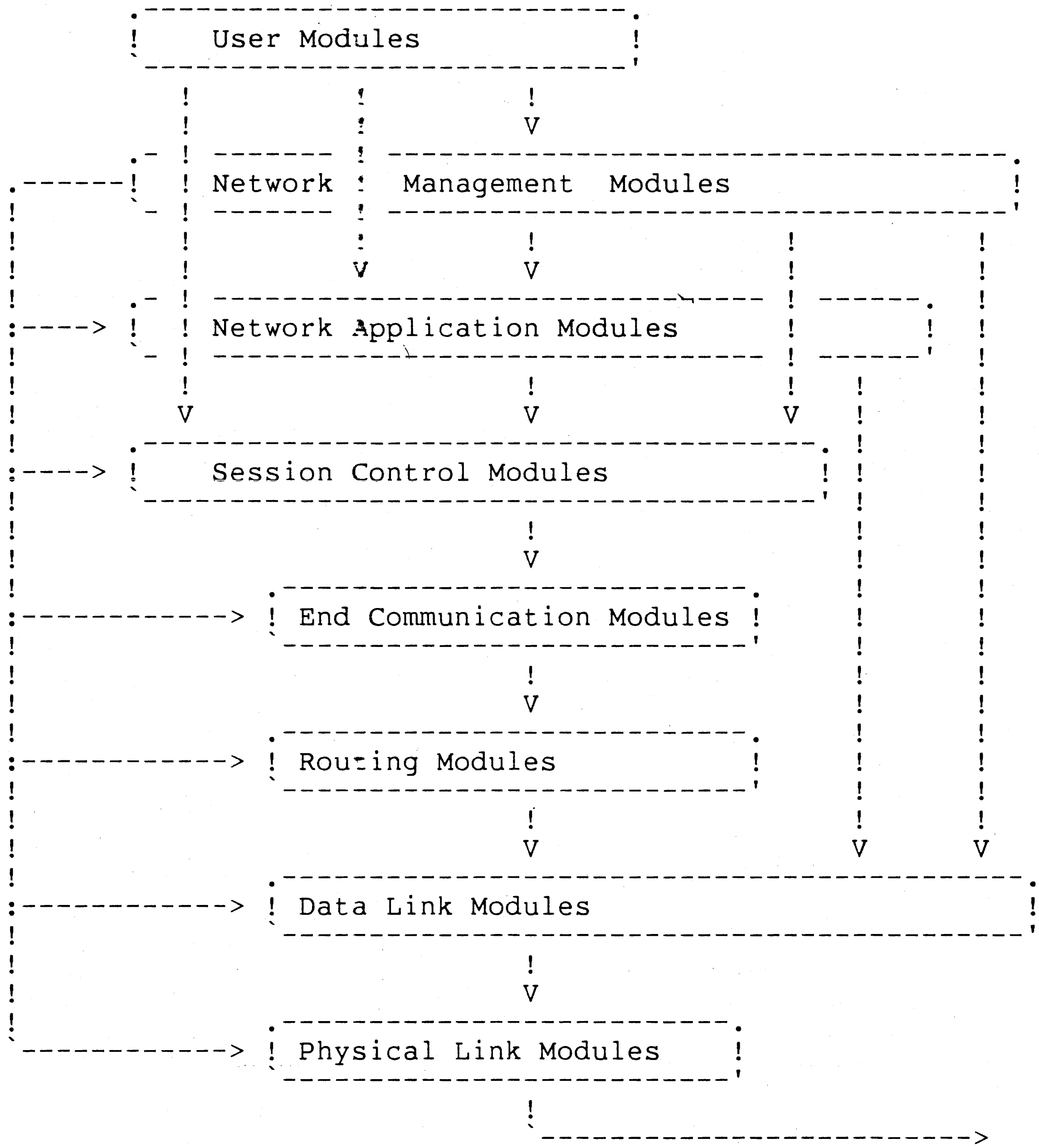


Figure 1. Network Management Relation to DNA

Network Management contains two models:

1. A simplified network model that is intended for network management use. This model is in a sense a simplified view of DNA (Section 2.3).

2. The model for Network Management as part of DNA (Section 2.4).

2.3 Network Management Model of DNA from the User Perspective

Because one of the primary goals of the Network Management design is ease of use, the person who uses the Network Management software is presented with a different, less complicated view of the network than that of the entire DNA model. This model is addressed at two levels: the interactive user at a terminal and the user program.

The interactive user manages the network mainly by entering commands of the form:

verb entity entity-option

The verb is an English verb such as SET, CLEAR, SHOW, LOAD, or LOOP. The entity is one of five classes of controllable network elements:

1. Node - Each node in a network represents a distinct operating system with associated CPU and peripherals. Nodes are further described in the section entitled Nodes.

2. Area - An area is a group of nodes. Nodes are grouped into areas for hierarchical routing purposes.

3. Logging - Logging is the mechanism that keeps track automatically of important aspects of the network operation. Logging is further explained in Section 2.3.3.

4. Circuits - Circuits are logical communications paths described in Section 2.3.4.

5. Lines - Lines are physical communications paths described in Section 2.3.4.

6. Modules - Modules are any entity that does not fit into the above classifications but represents a distinct function and/or database. For the present, all the modules are related to either X.25 or maintenance functions.

Note that in particular implementations, the word "component" or "element" may be used in place of "entity".

The user can observe, and, in some cases, control various aspects of entities. The entity-option qualifies the aspect of the entity upon which the verb is to act. For each entity, DNA specifies associated parameters. For some entities DNA also specifies counters and events. The parameters and counters are information kept in data bases.

Data bases are of two types: volatile and permanent. The volatile data base describes the running network. If the system crashes or shuts down, the volatile data base disappears. The permanent data base specifies the initial content of the volatile data base. Counters are only kept in the volatile data base.

Events are not kept in any data base. Events are captured by the event logging mechanism as they occur.

Parameters are values in a data base indicating characteristics or status of an entity. For example, some of the node parameters are:

NODE STATE
NODE NAME
NODE ADDRESS

Some parameters can be changed or set. Of these, some can be cleared to a default value or to no operation. Parameters can be read.

Counters are variables kept in main memory. Examples of circuit counters are:

Seconds since last zeroed
Bytes received
Bytes sent

Counters can only be read, zeroed, or read and zeroed.

Events are occurrences in the network that the Network Management event logging mechanism keeps track of. Only the logging entity has events. Examples of logging events are:

Invalid message
Verification reject
Line counters zeroed
Node reachability change

The user cannot directly control events. The user can, however, control aspects of the logging of events.

The user program interface uses specified messages to pass the same types of requests as the interactive user can make.

2.3.1 Nodes

Nodes are the major controllable entity of the network. They are the addressable objects of the Routing algorithms. From the standpoint of Network Management, there are two major classifications of nodes: executor and remote. The executor is the node actually executing a

Network Management function. All other nodes in relation to the executor are remote. Network concepts and functions described in this document are described from the vantage point of the executor node.

Note that from the point of view of a user, the terminal he is using is at the local node. This is usually also the executor node. However, if this user should set his terminal as a virtual terminal to access Network Management functions at another node, then his "physically" local node is a "remote" node from the point of view of Network Management.

In some contexts, nodes are also referred to as loop, command, host, or target nodes.

Loop node is a special name for the executor node. This name is associated with one of the executor's circuits, and logical links to that node are routed out the line with the expectation that they will be looped back.

The command node is the node requesting a high level Network Management function from the executor. It can be the executor itself or some remote node.

A host node is a node that provides some service, such as a file system.

A target node is the node that is to receive a load, loop back a low level line test message, or generate a dump.

In any particular operation, functions can be distributed among nodes. For example, a down-line system load can use different command, executor, host, and target nodes. Alternatively, the down-line load can use just the executor and target nodes; or executor, command and target; or executor, host, and target.

The node entity is associated with functions, parameters, counters, and events from the Network Management, Session Control, End Communication, and Routing layers of the general DNA model.

2.3.2 Areas

An area is a group of nodes. The Network Manager groups nodes into areas for hierarchical routing purposes. The use of areas in a network allows node identification within an area to be independent of node identification within other areas. Each area is uniquely identified. The addition of an area identification to a node identification uniquely identifies a node within the network. Nodes in a single area network will, by convention, have the default area number "1", which will not be displayed, thus hiding the unnecessary addressing hierarchy from the Network Manager.

2.3.3 Logging

Logging is the automatic event recording mechanism of Network Management. A logged event is directed to a sink node for output. A sink may be a system console, a file, or a monitor program. The node at which the event occurs is the source node. The system manager must use Network Management commands to tell the source node what kinds of events are to be logged and to what kinds of sinks. The sinks can be located at one or more nodes. These nodes are the destination nodes. The Network Management software at each destination node knows the actual name and state of its resident sinks.

Some examples of logging entity-options are:

LOGGING STATE
LOGGING SINK NODE
LOGGING EVENTS

Logging sink functions and parameters, other than the actual creation of event data, are completely within the Network Management layer.

2.3.4 Circuits and Lines

The circuit and line entities are presented together as a reflection of the close coupling of the Data Link and Physical Link layers.

A circuit is a high level communications path. Circuits provide logical communication between protocol handling modules. They are the communications paths that are visible, for example, to Routing and the X.25 Gateway server. A circuit may be a permanent or switchable connection. Unknown to its high level user, a circuit may be in one-to-one correspondence to a physical link, multiplexed with many other circuits, and/or traffic split over multiple physical links. Some characteristics of circuits can affect the way that they are used, so in many cases the higher level can or must be aware of these differences. In other words, the line to circuit mapping is invisible, but other characteristics may not be.

A line is a low level communications path. Lines provide physical communications. They are the media over which circuits operate.

There are currently three major classes of circuits -- DDCMP, X.25 and Ethernet. DDCMP circuits are subdivided into point-to-point, multipoint control, and multipoint tributary. X.25 circuits are subdivided into permanent and switched, with switched further subdivided into incoming and outgoing. X.25 circuit parameters are from X.25 level 3, the packet level. X.25 line parameters are from X.25 level 2, the frame level.

DDCMP point-to-point circuits have a one-to-one correspondence between the circuit and the line.

For DDCMP, each multipoint tributary is a separate circuit, and all of the tributaries in a group use the same line. The line must be of protocol type DDCMP CONTROL. In other words, at the master end, there is one DDCMP control line. It is associated with one or more circuits, each of which has its own physical tributary address. At the slave end, there is a one-to-one correspondence of circuit and a DDCMP tributary line.

X.25 circuits differ from DDCMP circuits in that there is no direct correspondence between circuit and line. All X.25 circuits pass through the X.25 protocol handler module. Lines belong to the protocol handler module, and it is responsible for establishment and maintenance of the circuits that use them.

X.25 permanent circuits are very similar to DDCMP circuits in that both have predefined end points that are assumed in the usage of the circuit.

X.25 switched circuits can only be individually named in the context of a higher level user, such as Routing. This provides a handle for higher level user parameters or counters. For other users, they have no individual existence that is visible to Network Management.

Ethernet circuits are rather different from the other types in that there is not a single node at the other end. Rather, Ethernet circuits are distinguished from one another according to the higher level user's protocol. An Ethernet circuit is a path to many nodes and the visibility of these nodes to Network Management varies according to the higher level user.

Use of an Ethernet circuit requires a station identification. This station identification is an Ethernet address. The address that the station is currently using is called the physical address. Some stations will also respond to a group identification called a multicast address. Some stations also have an address, or addresses, built into their hardware. This hardware address may sometimes be used as the physical address. DNA currently requires that stations cabable of anything other than maintenance operations use a physical address that is a function of the DNA node address. This requirement is found in the DNA Ethernet Node Product Architecture Specification.

Circuit functions, parameters, counters, and events are from the Network Management, Routing, Data Link, and Physical Link layers. Line functions, parameters, counters, and events are from the Network Management, Data Link, and Physical Link layers.

2.3.5 Modules

Modules currently comprise the access routines, server and protocol handler for X.25, and Network Management maintenance handlers.

The X.25 access routine module contains the data base needed to connect the program using the access routines to a server for the desired public data network. This data base is organized by network identification.

The X.25 server module contains the data base needed to map an incoming X.25 call to a DECnet process and form the connection. This data base is organized by destination identification. The server module also keeps one set of counters relative to its internal resources.

The X.25 protocol handler module contains the common data base needed to multiplex switched and permanent X.25 circuits over its line or lines. These parameters are from X.25 level 3, the packet level. This data base is organized by one or more local DTE (Data Terminal Equipment -- the X.25 equivalent of a node) addresses. The protocol handler module contains an X.25 user group data base organized by group name. The protocol handler module also keeps counters relative to each of its local DTE addresses.

The Network Management maintenance modules are responsible for handling maintenance functions on circuits and/or lines. They have implied responsibility for handling maintenance for all DDCMP and X.25 data links and specific responsibility for Ethernet circuits assigned to them by the network manager. The maintenance modules represent a simplification for the network manager. They actually cover parts of the Network Management Link Watcher and Data Link Service Functions described in a later section.

Each of the maintenance modules contains the low level data base necessary to perform their respective functions on Ethernet circuits. Within the modules, the information is organized by circuit.

The looper module is necessary for Ethernet loopback testing.

The loader module is necessary for Ethernet up-line dump and down-line load.

The console module is necessary for Ethernet remote console functions.

The configurator module is necessary for determining the list of stations on an Ethernet line. It is a user of the console module.

Module functions, parameters, counters, and events are currently from the Network Applications, Network Management, and Data Link layers.

2.4 Model of Network Management Components in the DNA Layers

The functional components of Network Management are as follows:

user layer components

Network Control Program (NCP). The Network Control Program enables the operator to control and observe the network from a terminal. Section 4 specifies the NCP commands.

Network Management layer components

Section 5 specifies the Network Management layer components and their operation. Figure 2, following, shows the relationship of Network Management layer modules in a single node.

The components are:

Network Management Access Routines. These routines provide user programs and NCP with generic Network Management functions, and either convert them to Network Information and Control Exchange (NICE) protocol messages or pass them on to the Local Network Management Function. Section 5.1 describes the Network Management Access Routine's operation.

Network Management Listener. The Network Management Listener receives Network Management commands from the Network Management level of remote nodes, via the NICE protocol. In some implementations it also receives commands from the local Network Management Access Routines via the NICE protocol. It passes these requests to the Local Network Management Function. Section 5.1 describes the Network Management Listener.

Local Network Management Functions. These take function requests from the Network Management Listener and the Network Management Access Routines and convert them to system dependent calls. They also provide interfaces to lower level modules directly for control purposes. Section 5.2 describes the Local Network Management Function's operation.

Link Watcher. The Link Watcher is a component in a node that can sense service requests on a data link from a physically adjacent node. it controls automatically-sensed down-line load or up-line dump requests. Section 5.3 describes the Link Watcher operation.

Maintenance Functions. These are the actual maintenance operations, such as down-line load or link loop test, that are specified in the DNA Low Level Maintenance Operation specification.

Data Link Service Functions. These provide the Link Watcher and the Local Network Management Functions with line services needed for service functions that require a direct interface to the data link layer (line level testing, down-line loading, up-line dumping, triggering a remote system's bootstrap loader and setting the line state). The Data Link Service software (or hardware) component maintains internal states as well as line substates. Section 5.4 describes the Data Link Service operation.

Event Logger. The Event Logger provides the capability of logging significant events for operator intervention or future reference. The process concerned with the event (for example, Routing) provides the data to the Event Logger, which can then record it. Section 5.5 describes the Event Logger operation.

Network Application layer components

Loopback Mirror. Access and service routines communicate using the Loopback Mirror Protocol to provide node level loopback on logical links. Section 5.13 describes this Network Application layer component.

Object Types

The Network Management architecture requires three separate object types. Each has a unique object type number.

The object types and numbers are:

Type	Object Type Number
Network Management Listener	19
Loopback Mirror	25
Event Receiver	26

Table 0 - Network Management Object Types

```
      .--------.`                          |  Interfaces for function
     / \________ .                         V  requests
    '   !.-----.  !
    `   !!     !  !----------.             ->  Control Interfaces
     \  `-----'--'.          |
      \ `%%%%%%\   \         V
       `==========='   .----------.       .-------------------.
                       |   NCP    |       |   User Program    |
                       `----------'       `-------------------'
  USER LAYER                     |        |
  ============================== | ====== | ===========================
  .---------.                    |        |        Network Management
  | Link    |                    |        |        commands from other
  | Watcher |                    V        V        nodes------------.
  `---------'                .------------.         .-----------.    |
    |  |                     | Network    |         | Network   |    |
    |  | Network       NICE  | Management | NICE    | Management|<--'
    |  | Management   Protocol| Access    | Protocol| Listener  |
    |  | commands<------------| Routines  |-------->|           |
    |  | to other            `------------'         `-----------'
    |  | nodes                     |                      |
    |  V                           V                      V
    | .-------------------------------------------------------------.
    | |               Local Network Management Functions            |
    | `-------------------------------------------------------------'
    |   |       |         |  |  |    |   |
    |   |       V         |  |  |    |   |
    |   |  .-------------.|  |  |    |   |
    |   |  | Maintenance |<-'  |  |    |   |
    |   |  | Functions   |     |  |    |   |
    |   |  `-------------'     |  |    |   |
    |   |      |               |  |    |   |
    V   V      V               |  |    |   |       Events from
  .------------.               |  |    |   | Events to   other nodes---.
  | Data Link  |               |  |    |   | other nodes               |
  | Service    |<--------------'  |    |   | <----------.-----------.  |
  | Functions  |                  |    |   |            | Event     |<-'
  `------------'                  |    |   `----------->| Logger    |
        |                         |    |                `-----------'
  NETWORK MANAGEMENT LAYER        |    |                 |       |
  ==== | ==================== | == | ================ | ====== | =====
  LOWER LAYERS                    |    |                 |       |
       |                          |    |                 |       |
       V                          |    V                 V       |
   Service interface to           | System dependent calls to    |
   Data Link Layer                | application layer and local  |
                                  | operating system functions   |
                                  | (file access, logical link   |
      Control over lower    <----' loopback, set timer, etc.)    |
      level functions                                            V
      (examine line state,                        Control interface to
      turn on NSP, etc.)                          read event queues
```

Figure 2. Network Management Layer Modules and Interfaces in a Single Node.

3 NETWORK MANAGEMENT AS SEEN BY THE USER

This section describes in detail the entities and their related parameters, counters, events and other entity options. These descriptions are both an introduction to and a reference for the NCP commands described in Section 4. Section 4.1 describes the Network Management functions for NCP commands that use the parameters described in this section. User programs can access these functions using the NICE protocol (Section 6). The NICE protocol binary formats associated with the entities, parameters, counters, and events are specified in Section 7.

The descriptions in this section relate parameters, counters, and events to the architectural layers to which they belong. Some parameters are specified as read-only. This means that Network Management can only read the value, and cannot directly change it. Changes to parameter values are typically under control of the DNA layer that "owns" the parameter.

In some of the descriptions in this section the term id-string is used to describe an identification. In all of these cases, an id-string consists of one to sixteen characters from the set of upper-case alphabetics, numerics, period and hyphen. An id-string must contain at least one alphabetic character.

Ethernet addresses appear several times as parameters called ethernet-address. An ethernet-address is a string of 12 hexadecimal digits, bytes separated by hyphens. The bytes are ordered from left to right as transmitted and received on the Ethernet. An example address is AA-00-04-00-0E-01.

In the following descriptions keywords (words that Network Management reserves for use in NCP commands) are capitalized. Brackets enclose optional input or output.

3.1 Nodes

A node is an implementation of a computer system that supports Routing, End Communication, and Session Control. Each node has a unique address assigned by the manager of each node. The Routing layer sends user data to nodes according to the node address. Since it is easier for humans to address nodes by names, DNA allows one node name for each node. The network manager should make sure that each node name and address in the network is unique. (An implementation may also provide the ability to assign additional node names to nodes, but these names can be known to the local node only. Refer to the Routing specification.)

The user can identify nodes in two major ways: individually and in groups. To identify a node individually, there are, again, two major ways:

1. Specify the keyword NODE along with a node-identification in

the format:

NODE node-identification

The node-identification is one of the following:

node-name — A node name consists of one to six upper case alphanumeric characters with at least one alpha character. A node name must be unique within a node and should be unique within the network.

node-address — A node address is a hierarchically structured number assigned to a particular node. It consists of two parts, an area number and a node number. Each of these is an unsigned decimal integer. They are displayed or entered separated by a single period. If the area number is not specified, it defaults to the area of the executor node. Each node address must be unique. Node numbers must be unique within an area, but may be re-used in different areas. Only the address of the executor node can be set. Note: In a single area network, the area number defaults to "1" (by convention), and is hidden from the user.

2. Specify the executor node with the keyword EXECUTOR, as follows:

   EXECUTOR

Node group identifications are as follows:

ACTIVE NODES — For a nonrouting node, all nodes that the executor sees on the other end of a logical link, or as adjacent, or as designated router. For an intra-area routing node, all of the above plus all nodes that the executor perceives as reachable within its area. For an inter-area router, all of the above plus all nodes the executor sees as adjacent inter-area routers.

ADJACENT NODES — All nodes that the executor perceives Routing can reach and that are physically adjacent (i.e. separated from the executor by a single circuit). Each occurrence of a node on a different circuit appears as a separate adjacent node. In other words, the adjacency of a node is qualified by the circuit on which it is adjacent.

KNOWN NODES — As defined for ACTIVE NODES, plus all nodes that have a name, including names that map to a circuit (i.e., loop nodes).

LOOP NODES — All nodes that are associated with a circuit for loop testing purposes.

SIGNIFICANT NODES — All nodes that have significant information associated with them for display purposes.

When obtaining node information, that pertaining to the executor node is returned first, and that for loop nodes, last.

The format for displaying node identification is:

NODE = node-address [(node-name)]

For example:

NODE = 19 (Elrond)

NODE = 3.19 (Fargon)

A node-related routing concept that is visible in Network Management is that of an adjacency. An adjacency is an adjacent node as reachable over a particular circuit. Each different circuit that leads to an adjacent node counts as a separate adjacency. Each different adjacent node on a circuit counts as a separate adjacency.

3.1.1 Node Parameters

The node parameters following are listed in alphabetical order according to the DNA layer that owns them, starting with the highest layer that contains node parameters.

The executor node keeps two data bases relative to nodes:

1. A data base of its own node parameters

2. A data base of remote node parameters (for each remote node) and of optional adjacent node parameters.

Many types of parameters kept in the executor node data base are not kept in the data bases the executor keeps for remote nodes. Also, some remote node parameters can only be kept for adjacent nodes.

Thus, in the descriptions below some parameters are distinguished as applying to the executor node, remote node, or the adjacent node.

Some parameters are described as loop-only. This means that they are parameters that only exist for use with the LOOP command and the Test message. Some of them have fixed, default values.

3.1.1.1 Network Management Layer

COUNTER TIMER seconds

This value is the number of seconds between node counter log events. The expiration of the timer causes a node counter logging event. Refer to the two sections entitled "Node Counters" and "Events" for lists of node counters and events. When the counter timer expires, the node counters are recorded as data in the event and then zeroed. If no value is set, node counters are not automatically logged. Seconds is specified as a decimal number in the range 1-65535.

CPU cpu-type

This value indicates the default target node CPU type for down-line loading the adjacent node. The possible values are:

PDP8
PDP11
DECSYSTEM1020
VAX

DIAGNOSTIC FILE file-id

This is the identification of the file to read from when the adjacent node is down-line loaded and has requested "diagnostics". The file identification is a string that is interpreted depending on the file system of the executor.

DUMP ADDRESS octal-number

This value represents the address in memory to begin an up-line dump of the adjacent node.

DUMP COUNT number

This value is the default number of memory units to up-line dump from the adjacent node.

DUMP FILE file-id

This value is the identification of the file to write to when the adjacent node is up-line dumped. The file identification is a string that is interpreted according to the file system of the executor.

HARDWARE ADDRESS ethernet-address

This value is the Ethernet hardware address of the adjacent node. It is the Ethernet address that is assigned to the node system hardware. This address is necessary for communication with the system for such purposes as down-line load before it has been able to meet the DNA requirements for setting its physical address.

HOST node-id

For the executor node, this value identifies the node from which the executor receives its services. This value defaults to the executor node.

For adjacent nodes, this value is the host identification that the adjacent node receives when it is down-line loaded. This value defaults to the executor node.

IDENTIFICATION string

This is a text string that describes the executor node (for example, "Research Lab"). The string is 32 characters of any type. When entered in NCP, if the string contains blanks or tabs, it must be enclosed in quotation marks. A quotation mark within a quoted string is indicated by two adjacent quotation marks ("").

LOAD FILE file-id

This is the identification of the file to read from when the adjacent node is down-line loaded and has requested "operating system". The file identification is a string that is interpreted depending on the file system of the executor.

LOOP ASSISTANT NODE node-id

This identifies the loop-only parameter used only as input on the LOOP CIRCUIT command for Ethernet third-party circuit loop testing. This parameter applies to the executor node only.

LOOP ASSISTANT PHYSICAL ADDRESS ethernet-address

This is the loop-only parameter used only as input on the LOOP CIRCUIT command for Ethernet third-party circuit loop testing. It cannot be a multicast address. This parameter applies to the executor node only.

LOOP COUNT count

This is the loop-only default count for the number of times to loop the data for a loop test. This parameter applies to the executor node only. Its value is 1.

LOOP HELP help-type

This is the loop-only default help type for Ethernet circuit loop testing. This parameter applies to the executor node only. Its value is FULL.

LOOP LENGTH length

This is the loop-only default length for the data that is looped in a loop test. This parameter applies to the executor node only. Its value is 40.

LOOP NODE node-id

This identifies the loop-only parameter used only as input on the LOOP CIRCUIT command for Ethernet circuit loop testing. This parameter applies to the executor node only.

LOOP WITH block-type

This is the loop-only default block type for loop testing. This parameter applies to the executor node only. Its value is MIXED.

PHYSICAL ADDRESS ethernet-address

This read-only executor parameter is the Ethernet address currently in use to identify itself.

MANAGEMENT VERSION n.n.n

This is the read-only Network Management Version, consisting of the version number, the Engineering Change Order (ECO) number, and the user ECO number (for example, 3.0.0). This parameter applies to the executor node only.

SECONDARY DUMPER file-id

This identifies the secondary dumper file for up-line dumping the adjacent node. The file identification is interpreted depending on the file system of the executor.

SECONDARY LOADER file-id

This identifies the secondary loader file for down-line loading the adjacent node.

SERVICE CIRCUIT circuit-id

This identifies the circuit to the adjacent node for down-line loading and up-line dumping. This is the default parameter if the circuit-id is not included in a down-line load command.

SERVICE DEVICE device-type

This is the identification of the device type that the adjacent node uses for service functions when in service slave mode (Section 5.4). The device type is one of the standard line device mnemonics (Table 10, Section 7.4).

SERVICE NODE VERSION node-version

This is the DNA version of the adjacent node, which is used to determine the TARGET SYSTEM ADDRESS parameter in the MOP Parameter Load With Transfer Address message (see <u>DNA Maintenance Operations Functional Specification</u>). The default value is 1

(Phase IV).

SERVICE PASSWORD password

This is the password required to trigger the bootstrap mechanism on the adjacent node. The password is a hexadecimal number in the range 0 - FFFFFFFFFFFFFFFF (64 bits).

SOFTWARE IDENTIFICATION software-id

This identifies the software to be loaded when the adjacent node is down-line loaded. Software-id is a string of 1-16 characters.

SOFTWARE TYPE program-type

This value represents the target node initial software program type for down-line loading the adjacent node. Program type is one of:

SECONDARY [LOADER]
TERTIARY [LOADER]
SYSTEM

STATE node-state

This represents the operational state of the executor node. The possible states are:

ON	Allows logical links.
OFF	Allows no new links, terminates existing links, and stops routing traffic through.
SHUT	Allows no new logical links, does not destroy existing logical links, and goes to the OFF state when all logical links are gone.
RESTRICTED	Allows no new incoming logical links from other nodes.

TERTIARY LOADER file-id

This identifies the tertiary loader file for down-line loading the adjacent node. The file identification is interpreted according to the executor node file system.

3.1.1.2 Session Control Layer

ADDRESS node-address

This value is the address of the executor node. This parameter applies to the executor node only.

CIRCUIT circuit-id

This value identifies a loop node for testing and sets the identification of the circuit to be used for all traffic to the node. The circuit-id can be associated with only one loop node name. Refer to the section entitled Testing Link and Network.

INCOMING TIMER seconds

This value represents the maximum duration between the time a connect is received for a process at the executor node and the time that process accepts or rejects it. If the connect is not accepted or rejected by the user within the number of seconds specified, Session Control rejects it for the user. If no value is set, there is no timer.

NAME node-name

This parameter represents the name to be associated with the node identification. Only one name can be assigned to a node address or a circuit identification. No name should be used more than once in a network. Node-name is one to six upper case alphanumeric characters with at least one alpha character.

OUTGOING TIMER seconds

This value represents the duration between the time the executor requests a connect and the time that connect is acknowledged by the destination node. If the connect is not acknowledged within the number of seconds specified, Session Control returns an error. If no value is set, there is no timer. The range is 1-65535.

3.1.1.3 End Communication Layer

ACTIVE LINKS number

This read-only parameter represents the number of active logical links from the executor to the destination node.

DELAY seconds

This read-only parameter is the average round trip delay in seconds to the destination node. This parameter is kept on a remote node basis.

DELAY FACTOR number

This is the number by which to multiply one sixteenth of the estimated round trip delay to a node to set the retransmission timer to that node. The round trip delay is used in an NSP algorithm that determines when to retransmit a message (End Communication specification). The number is decimal in the range

1-255.

DELAY WEIGHT number

This number represents the weight to apply to a current round trip delay estimate to a remote node when updating the estimated round trip delay to a node. The number is decimal in the range 1-255. On some systems the number must be 1 less than a power of 2 for computational efficiency (End Communication specification).

INACTIVITY TIMER seconds

This value represents the maximum duration of inactivity (no data in either direction) on a logical link before the node checks to see if the logical link still works. If no activity occurs within the minimum number of seconds, End Communication generates artificial traffic to test the link (End Communication specification). The value range is 1-65535.

MAXIMUM LINKS number

This value represents the maximum active logical link count allowed for the executor. The count is a decimal number in the range 1-65535.

NSP VERSION n.n.n

This read-only parameter represents the version number of the node End Communication. The format is the same as for the Network Management version.

RETRANSMIT FACTOR number

This value represents the maximum number of times the source End Communication at the executor node will restart the retransmission timer when it expires. If the number is exceeded, Session Control disconnects the logical link for the user (End Communication specification). The number is decimal in the range 1-65535.

3.1.1.4 Routing Layer

AREA MAXIMUM COST number

This value represents the maximum total path cost allowed from the executor to any other level 2 routing node. The AREA MAXIMUM COST number is decimal in the range 1-1022. This parameter is only applicable if the executor node is of type AREA.

AREA MAXIMUM HOPS number

This value represents the maximum number of routing hops allowable from the executor to any other level 2 routing node.

The AREA MAXIMUM HOPS number is decimal in the range 1-30. This parameter is only applicable if the executor node is of type AREA.

BROADCAST ROUTING TIMER seconds

This value determines the maximum time allowed between Routing updates on Ethernet circuits. When this timer expires before a routing update occurs, a routing update is forced. With a standard calculation, Routing also uses this timer to enforce a minimum delay between routing updates. Seconds is a decimal integer in the range 1-65535.

BUFFER SIZE bytes

This parameter value determines the maximum size of a Routing message. It therefore determines the maximum size message that can be forwarded. The size is a decimal integer in the range 1-65535. This size is in bytes. This size includes protocol overhead down to and including the End Communication layer, plus a constant value of 6. (This value of 6 is included to provide compatibility with the parameter definition in Phase III, which included the Routing overhead.) It does not include Routing or Data link overhead (except for the constant value of 6). There is one buffer size for all circuits.

NOTE

> The BUFFER SIZE defines the maximum size messages that the Routing layer can forward. The SEGMENT BUFFER SIZE (defined below) defines the maximum size messages that the End Communication layer can transmit or receive. The SEGMENT BUFFER SIZE is always less than or equal to the BUFFER SIZE. Normally the two parameters will be equal. They may be different to allow the network manager to alter buffer sizes on all nodes without interruption of service. They both include an extra 6 bytes for compatibility with Phase III.

CIRCUIT circuit-id

This read-only parameter identifies the circuit used to get to a remote node. Circuit-id is an id-string.

This parameter can be used when displaying a list of nodes to indicate that the display is to be restricted to those nodes adjacent on the specified circuit.

COST cost

This read-only parameter represents the total cost over the current path to the destination node. Cost is a positive integer value associated with using a circuit. Routing routes messages (data) along the path between two nodes with the smallest cost.

COST is kept on a remote node basis.

HOPS hops

This read-only parameter represents the number of hops over to a destination node. A hop is Routing value representing the logical distance between two nodes in a network. HOPS is kept on a remote node basis.

MAXIMUM ADDRESS number

This value represents the largest node number and, therefore, number of nodes that can be known about by the executor node's home area. The number is an integer in the range 1-1023.

MAXIMUM AREA number

This value represents the largest area number and, therefore, number of areas that can be known about by the executor node's Routing. This parameter is only applicable if the executor node is of type AREA. The number is an integer in the range 1-63.

MAXIMUM BROADCAST NONROUTERS number

This value represents the maximum total number of nonrouters the executor node can have on its Ethernet circuits. The number is an integer in the range 0-65535.

MAXIMUM BROADCAST ROUTERS number

This value represents the maximum total number of routers the executor node can have on its Ethernet circuits. The number is an integer in the range 0-65535.

MAXIMUM BUFFERS number

This value represents the maximum number of transmit buffers that Routing may use for all circuits. The number is a decimal integer in the range 1-65535.

MAXIMUM CIRCUITS number

This value represents the maximum number of Routing circuits that the executor node can know about. The number is decimal in the range 1-65535.

MAXIMUM COST number

This value represents the maximum total path cost allowed from the executor to any node within an area. The path cost is the sum of the circuit costs along a path between two nodes. This parameter defines the point where the executor node's Routing routing decision algorithm declares another node unreachable because the cost of the least costly path to the other node is excessive. For correct operation, this parameter must not be less than the maximum path cost of the network. The MAXIMUM COST number is decimal in the range 1-1022.

MAXIMUM HOPS number

This value represents the maximum number of routing hops allowable from the executor to any other reachable node within an area. (A hop is the logical distance over a circuit between two adjacent nodes.) This parameter defines the point where the executor node's Routing routing decision algorithm declares another node unreachable because the length of the shortest path between the two nodes is too long. For correct operation, this parameter must not be less than the network diameter. (The network diameter is the reachability distance between the two nodes of the network having the greatest reachability distance, where reachability distance is the length of the shortest path between a given pair of nodes.) The MAXIMUM HOPS number is decimal in the range 1-30.

MAXIMUM VISITS number

This value represents the maximum number of nodes a message coming into the executor node can have visited. If the message is not for this node and the MAXIMUM VISITS number is exceeded, the message is discarded. The MAXIMUM VISITS parameter defines the point where the packet lifetime control algorithm discards a packet that has traversed too many nodes. For correct operation, this parameter must not be less than the maximum path length of the network. (The maximum path length is the routing distance between the two nodes of the network having the greatest routing distance, where routing distance is the length of the least costly path between a given pair of nodes.) The MAXIMUM VISITS number is decimal in the range MAXIMUM HOPS to 63.

NEXT NODE node-id

This read-only value indicates the next node on the circuit used to get to the node under scrutiny.

ROUTING TIMER seconds

This value determines the maximum time allowed between Routing updates on non-Ethernet circuits. When this timer expires before a routing update occurs, a routing update is forced. Seconds is a decimal integer in the range 1-65535.

ROUTING VERSION n.n.n

This read-only parameter identifies the executor node's Routing version number. The format is the same as for the Network Management version number.

SEGMENT BUFFER SIZE bytes

This parameter value determines the maximum size of an end-to-end segment. The size is a decimal integer in the range 1-65535. This size is in bytes. This size includes protocol overhead down to and including the End Communication layer, plus a constant value of 6. (This value of 6 is included to provide compatibility with the BUFFER SIZE parameter definition.) It does not include Routing or Data link overhead (except for the constant value of 6). See additional note for BUFFER SIZE.

SUBADDRESSES subaddress-range

This parameter is the range of local DTE subaddresses that are acceptable on any X.25 circuit for an incoming call. Subaddress-range consists of either a single subaddress or two subaddresses separated by only a hyphen. A subaddress is a decimal integer in the range 0-9999. If two subaddresses are provided, the second must be greater than the first.

TYPE node-type

This parameter indicates the type of the executor node. The node-type is one of the following:

ROUTING III
NONROUTING III
ROUTING IV
NONROUTING IV
AREA

A routing node has full routing capability. A nonrouting node contains a subset of the Routing routing modules. The III and IV indicate the DNA phase of the node. Nonrouting nodes can deliver and receive packets to and from any node, but cannot route packets from other nodes through to other nodes. An area node routes between areas. Refer to the Routing specification for details.

For adjacent nodes, this is a read-only parameter that indicates the type of the reachable adjacent node.

3.1.2 Node Counters

Network Management displays or zeroes node counters as a group. The following Network Management counter is kept for nodes:

Seconds since last zeroed

The following End Communication counters are kept for nodes:

User bytes received
User bytes sent
User messages received
User messages sent
Total bytes received

Total bytes sent
Total messages received
Total messages sent
Connects received
Connects sent
Response timeouts
Received connect resource errors
Maximum logical links active (executor only)

The following Routing counters are kept for the executor node:

Aged packet loss
Node unreachable packet loss
Node out-of-range packet loss
Oversized packet loss
Packet format error
Partial routing update loss
Verification reject

Refer to the relevant specifications for further explanation of the type of information counted.

3.2 Areas

An area is a group of nodes. The network manager groups nodes for hierarchical routing purposes. (Refer to the Routing specification.)

The user can identify areas in two major ways: individually and in groups. To identify an area individually, use the area number. An area number is a decimal integer in the range 1-63. By convention, Area "1" is used to designate a single area network and Area "0" is used when communicating with a Phase III node.

Area group identifications are as follows:

ACTIVE AREAS	All areas that the executor perceives Routing can reach.
KNOWN AREAS	Same as ACTIVE AREAS.

All of the area parameters are owned by the routing layer and are as follows:

CIRCUIT circuit-id

This read-only parameter identifies the circuit used to get to a remote area. Circuit-id is an id-string.

COST cost

This read-only parameter represents the total cost over the current path to the destination area. Cost is a positive integer value associated with using a circuit. Routing routes messages (data) along the path between two areas with the smallest cost.

HOPS hops

This read-only parameter represents the number of hops over to a destination area. A hop is Routing value representing the logical distance between two areas in a network.

NEXT NODE node-id

This read-only value indicates the next node on the circuit used to get to the area under scrutiny.

STATE state

This read-only value indicates the state of the area, either REACHABLE, or UNREACHABLE.

3.3 Logging

Logging is the Network Management automatic event-recording mechanism.

The logging entity identification is the sink type. Logging may be referred to by individual sink types or by the sink types as a group. The formats for specifying logging entities symbolically are as follows:

LOGGING sink-type	A particular logging sink type
KNOWN LOGGING	All logging sink types known to the executor node
ACTIVE LOGGING	All known sink types that are in ON or HOLD state
SIGNIFICANT LOGGING	All known sink types that have significant information for display purposes.

A sink type is one of the following:

CONSOLE
FILE
MONITOR

Network Management sends information about logged events to sink nodes. The user establishes sink nodes with NCP commands. Sink node identification is as follows:

SINK NODE node-identification
or
SINK EXECUTOR

The default sink-node is the executor node.

3.3.1 Source Qualifiers

Events occur at logging sources. Since logging for a specific entity can be different from logging for that entity as a group, the user can specify that specific sources be logged by using the source-qualifier option. Source-qualifier can be one of the following:

```
AREA area-id
CIRCUIT circuit-id
LINE line-id
MODULE module-id
NODE node-id
```

Refer to Sections 3.1, 3.4, 3.5, and 3.7 for descriptions of node-id, circuit-id, line-id, and module-id.

3.3.2 Logging Parameters

All the logging parameters are owned by the Network Management layer. These parameters are as follows:

EVENTS event-list

This set of values indicates the types and classes of events to be recorded at the sink-node. Tables 22 and 23, Section 7.12, specify event classes and types. Event-list consists of event class.event type(s). The types are specified in ranges using hyphens, in lists using commas, or a combination of both. Examples of event-lists are:

```
3.0-2
4.1-4,8,10
6.1,3,5
```

wild card notation indicates all types of events for a particular class. For example,

```
3.*
```

The keywords KNOWN EVENTS can replace EVENTS event-list in NCP commands. KNOWN EVENTS imply all events known to the executor node for the specified sink node and source. If no source is specified, source specific events are not affected.

NAME sink-name

This is the identification of the executor node's logging sink. Sink-name has one of three forms depending on the sink-type:

Type	Sink-name
CONSOLE	device-id
FILE	file-id
MONITOR	process-id

The sink name format depends on what the executor system understands.

SINK NODE node-id

This parameter identifies the sink node to which the other parameters in a command or response apply. The default sink node is the executor. Node-id is either a node name or a node address (Section 3.1).

STATE sink-state

This value indicates the executor node's logging state for the sink type. The possible values of sink-state are:

ON The sink is available for receiving events.

OFF The sink is not available and any events destined for it should be discarded.

HOLD The sink is temporarily unavailable and events should be queued.

There are no logging counters. Section 3.8 describes event parameters.

3.4 Circuits

Circuits are logical communications paths providing communications between adjacent nodes. A circuit may be identical to a physical link, multiplexed with many other circuits, and/or traffic split over multiple physical links.

Circuit identification is a circuit name with the format of an id-string. Network Management keeps a master list of circuit names, ensuring their uniqueness for the Data Link layer.

Circuits can be identified in groups as follows:

KNOWN CIRCUITS - All circuits that have a name.

ACTIVE CIRCUITS - All circuits in the ON or SERVICE state.

SIGNIFICANT CIRCUITS - All circuits that have at least one parameter.

A circuit can have an owner. This means that the circuit is reserved for the exclusive use of the owner. For example, the owner may be the executor node (Routing) or some other network component.

Whether or not it has an owner, a circuit has a user whenever it is open for use through the mechanisms of the Data Link interface. User in this sense is a network component, not a person. Currently, the user can be either the owner or the X.25 protocol module.

When Network Management uses a circuit, the user's rights are overridden. For example, Network Management must take over the circuit to execute such service functions as down-line loading and loop testing. When Network Management finishes its prescribed function, the circuit is returned to the user.

3.4.1 Circuit Parameters

There are five groups of circuit parameters:

1. Common circuit parameters -- These are parameters common to all circuits (Refer to the section entitled Common Circuit Parameters).

2. Executor node circuit parameters -- These are parameters that apply only to circuits whose owner is the executor node. (Refer to the section entitled Executor Node Circuit Parameters.)

3. DDCMP circuit parameters -- These are parameters that apply to DDCMP circuits only. (Refer to the section entitled DDCMP Circuit Parameters.)

4. X.25 circuit parameters -- These parameters apply to X.25 circuits only (Refer to the section entitled X.25 Circuit Parameters.)

5. Ethernet circuit parameters -- These parameters apply to Ethernet circuits only (Refer to the section entitled Ethernet Circuit Parameters.)

3.4.1.1 Common Circuit Parameters

The following parameters are common to all circuits:

COUNTER TIMER seconds

> This value represents the number of seconds the Network Management counter timer will run. The expiration of the counter timer causes a circuit counter logging event. The types of counters logged depends on the circuit protocol. Circuit counters are described in Section 3.4.2. The circuit counters

are recorded as data in a logging event and then zeroed. If no counter timer value is set, the circuit's counters are not automatically logged. Seconds is a decimal integer in the range 1-65535.

OWNER owner-id

This value identifies the circuit owner. Except for overrides through Network Management, the owner has exclusive rights to use the circuit. If no owner value is set, the circuit is available on a first-come, first-served basis.

To use a circuit, the owner must open it according to the rules of the particular Data Link interface. Ownership of a circuit has no implication as to whether the circuit is actively open by its owner or any other process. Setting the OWNER parameter merely reserves the circuit.

An owner-id consists of an entity type and entity identification. The executor node can be owner of any circuit. This implies that the circuit is actually reserved the DECnet routing module. Ethernet circuits can be owned by MODULE LOOPER, CONSOLE, or LOADER. These are circuits over which the management module can perform such management functions as loop tests or down-line loads.

From the standpoint of Network Management, Ethernet circuits automatically take on the proper Ethernet protocol types and multicast addresses according to their owner's requirements.

STATE circuit-state

This value represents the circuit's Network Management operational state as described in the state and substate model presented in Section 3.6.

SUBSTATE

This is the circuit's read-only Network Management substate (Section 3.6).

TYPE circuit-type

This value represents the type of the circuit. For X.25 circuits, the value must be set to X25. For DDCMP and Ethernet circuits it is read only and is the same value as the protocol of the associated line (see PROTOCOL in section entitled Common Line Parameters).

USER user-id

This is the read-only identification of the active user of the circuit. It tells the network manager what module is using the circuit.

In the case of a circuit with an owner, the user is the owner, but only when the owner has the circuit open. In the case of a circuit with no owner, the user is the network component that opened the circuit.

A user-id consists of an entity type and entity identification. The only user-ids currently defined are EXECUTOR and MODULEs X25-SERVER, LOOPER, LOADER, CONSOLE, and CONFIGURATOR.

3.4.1.2 Executor Node Circuit Parameters

The following parameters apply to circuits that are owned by the executor node:

ADJACENT NODE node-id

This read-only value indicates an adjacent node on the circuit. For Ethernet circuits there can be many adjacent nodes. This parameter can be used when displaying a list of circuits to indicate that the display is to be restricted to those circuits leading to the specified adjacent node.

BLOCK SIZE byte-count

This read-only parameter is the block size that was negotiated with the adjacent Routing layer during Routing initialization over a particular circuit. It includes the routing header, but excludes the data link header. This parameter is qualified by ADJACENT NODE.

COST cost

This value represents the Routing routing cost of the circuit. The cost is a decimal integer in the range 1-25. Routing routes messages along the path between two nodes having the smallest cost.

HELLO TIMER seconds

This value determines the frequency of Routing Hello messages sent to the adjacent node on the circuit. Seconds is a decimal integer in the range 1-8191.

LISTEN TIMER seconds

This read-only value determines the maximum time allowed to elapse before Routing receives some message (either a Hello message or a user message) from the adjacent node on the circuit. It was agreed during Routing initialization with the adjacent Routing layer. Seconds is a decimal integer in the range 1-65535. This parameter is qualified by ADJACENT NODE.

LOOPBACK NAME node-name

This parameter is the Session Control node name associated with a circuit as a result of the "SET NODE node-id CIRCUIT circuit-id" command. From the circuit standpoint, this is a read-only parameter.

ORIGINATING QUEUE LIMIT queue-size

This parameter indicates the maximum number of originating packets that may be outstanding on this circuit. This does not include route-thru traffic.

RECALL TIMER seconds

This parameter represents the minimum number of seconds to wait before restarting the circuit. If no value is set, there is no wait. Seconds is a decimal integer in the range 1-65535.

3.4.1.3 DDCMP Circuit Parameters

DDCMP circuits support the Network Management service functions of dump, load, and active and passive circuit loopback. The following parameters apply to DDCMP circuits:

LINE line-id

This value is the Data Link layer identification of the line that is to be used for traffic on the circuit. Line-id is a line name (Section 3.5).

SERVICE service-control

This value indicates whether or not Network Management allows service operations on a circuit. The values for service-control are as follows:

ENABLED	SERVICE state and/or service functions are allowed.
DISABLED	SERVICE state and/or service functions are not allowed.

TRIBUTARY tributary-address

This value represents the Data Link physical tributary address of the circuit. The tributary address is a decimal integer in the range 1-255.

The following parameters apply to DDCMP CONTROL circuits. In those cases where a value is specified in milliseconds, there is no assumption that all implementations can provide such fine resolution.

ACTIVE/INACTIVE/DYING BASE base

This value represents the base priority to which a tributary is reset each time it has been polled. A separate base can be set for each of the indicated polling states. Base is a decimal integer in the range 0-255. If not set, the defaults are: active, 255; inactive, 0; and dying, 0.

ACTIVE/INACTIVE/DYING INCREMENT increment

This value represents the increment added to the tributary priority each time the scheduling timer expires. Increment is a decimal integer in the range 0-255. If not set, the defaults are: active, 0; inactive, 64; and dying, 16.

BABBLE TIMER milliseconds

This value represents the number of milliseconds that a selected tributary or remote half-duplex station is allowed to transmit. Milliseconds is a decimal integer in the range 1-65535. If not set, the default is 6000 (6 seconds).

DEAD THRESHOLD count

This value represents the number of times to poll the active, inactive, or dying tributary before changing its polling state to dead because of receive timeouts. Count is a decimal integer in the range 0-255. If not set, the default is 8.

DYING THRESHOLD count

This value represents the number of times to poll the active or inactive tributary before changing its polling state to dying because of receive timeouts. Count is a decimal integer in the range 0-255. If not set, the default is 2.

INACTIVE THRESHOLD count

This value represents the number of times to poll the active tributary before changing its polling state to inactive because of no data response. Count is a decimal integer in the range 0-255. If not set, the default is 8.

MAXIMUM BUFFERS count

This value represents the maximum number of buffers the tributary can use from a common buffer pool. If not set, there is no common buffer pool and buffers are explicitly supplied by the higher level. Count is a decimal integer in the range 1-254 or the keyword UNLIMITED.

MAXIMUM TRANSMIT count

This value represents the maximum number of data messages that can be transmitted at one time. Count is a decimal integer in the range 1-255. If not set, the default is 4.

POLLING STATE polling-state

This value represents the state of the tributary relative to the multipoint polling algorithm. If not set the default is AUTOMATIC. The possible states are:

AUTOMATIC

The tributary's state is allowed to vary according to the operation of the polling algorithm.

ACTIVE/INACTIVE/DYING/DEAD

The tributary is locked in the specified state.

Polling-substate

This value represents the tributary's state as determined by the polling algorithm. This applies only when the polling state is AUTOMATIC and is read-only to Network Management. Polling-substate is one of ACTIVE, INACTIVE, DYING, or DEAD. It is displayed as a tag on the polling state, for example:

AUTOMATIC-INACTIVE

TRANSMIT TIMER milliseconds

This value represents the number of milliseconds to delay between data message transmits. Milliseconds is a decimal integer in the range 0-65535. If not set, the default is 0.

3.4.1.4 X.25 Circuit Parameters

X.25 circuits do not support any of the Network Management service functions. The following parameters apply to X.25 circuits:

MAXIMUM DATA byte-count

For permanent circuits, this value represents the Data Link maximum X.25 data size allowed on the circuit. For switched circuits owned by the executor node, this value represents the size that Routing is to request from X.25 for the circuit. Byte-count is a decimal integer in the range 1-65535. It must be <= to the maximum data size allowed within the X.25 protocol module.

MAXIMUM WINDOW block-count

For permanent circuits, this value represents the Data Link maximum number of X.25 blocks outstanding on the circuit. For switched circuits owned by the executor node, this value represents the window size that Routing is to request from X.25 for the circuit. Block-count is a decimal integer in the range

1-255.

USAGE usage-type

This Data Link parameter defines the usage type of an X.25 circuit. The usage-type values are as follows:

INCOMING Used only for switched incoming calls. Useful only for circuits that are owned by the executor node.

OUTGOING Used only for switched outgoing calls. Useful only for circuits that are owned by the executor node.

PERMANENT Permanently connected to the same remote station, and does not need to be dynamically switched.

BLOCKING blocking-control

This parameter applies to X.25 circuits that are owned by the executor node. The value indicates whether or not Routing will block messages before they are sent over the circuit. The values for blocking-control are as follows:

ENABLED Perform blocking as possible.

DISABLED No blocking.

NUMBER call-number

This parameter applies to either incoming or outgoing X.25 circuits that are owned by the executor node. The value represents the Routing full remote DTE address used to receive calls and to call out on the circuit. Call-number is a decimal integer of one to sixteen digits.

MAXIMUM RECALLS retry-count

This parameter applies to outgoing X.25 circuits that are owned by the executor node. The value represents the maximum number of Routing automatic call retries. Retry-count is a decimal integer in the range 0-255. If no value is set, there is no maximum.

CHANNEL channel-number

This parameter is the Data Link X.25 logical channel number to be used in running the X.25 protocol on the circuit. This applies only to permanent circuits. A channel-number is a decimal integer in the range 0-4095.

DTE dte-address

This parameter is the Data Link X.25 local DTE address to which the circuit belongs. This applies only to permanent circuits. Dte-address is a decimal integer of one to sixteen digits.

CONNECTED NODE node-id

This parameter is the read-only Application module identification of the node on which the DECnet object using the circuit resides. Node-id is a standard Network Management node identification (Section 3.1). This parameter applies only to permanent X.25 circuits being used by module X25-SERVER.

CONNECTED OBJECT object-id

This parameter applies only to permanent X.25 circuits being used by module X25-SERVER. The read-only Application module value identifies the DECnet object using the circuit. Object-id is as described for module X25-SERVER.

3.4.1.5 Ethernet Circuit Parameters

Ethernet circuits support all of the Network Management service functions through circuits that are owned by MODULE LOOPER, CONSOLE, or LOADER. The following parameters apply to Ethernet circuits:

DESIGNATED ROUTER node-id

This read-only value is the Routing layer identification of the node that is to be used for routing on circuits that are owned by the executor node.

LINE line-id

This value is the Data Link layer identification of the line that is to be used for traffic on the circuit. Line-id is a line name (Section 3.5).

MAXIMUM ROUTERS number

This parameter is the maximum number of routers (other than the executor itself) allowed on the circuit by Routing for circuits that are owned by the executor node. Number is a decimal integer in the range 0-255.

ROUTER PRIORITY number

This parameter is the priority that this router is to have in the selection of designated router for the circuit on circuits that are owned by the executor node. Number is a decimal integer in the range 0-127. The default value is 64.

SERVICE PHYSICAL ADDRESS ethernet-address

This parameter indicates the Ethernet physical address of an adjacent node that is being serviced on this circuit. This parameter is a qualifier for SERVICE SUBSTATE.

SERVICE SUBSTATE

This is the circuit's read-only Network Management substate (Section 3.6). It identifies the kind of service being performed on the circuit. This parameter is qualified by SERVICE PHYSICAL ADDRESS.

3.4.2 Circuit Counters

Network Management displays or zeroes counters as a group when the executor node (Routing) owns the circuit. The following Network Management counter is kept for circuits:

Seconds since last zeroed

The following Routing counters are kept for all circuits:

Terminating packets received
Originating packets sent
Terminating congestion loss
Transit packets received
Transit packets sent
Transit congestion loss
Circuit down
Adjacency down
Initialization failure

The following Data Link counters are kept for DDCMP circuits:

Bytes received
Bytes sent
Data blocks received
Data blocks sent
Data errors inbound
Data errors outbound
Remote reply timeouts
Local reply timeouts
Remote buffer errors
Local buffer errors
Selection intervals elapsed
Selection timeouts

The following Routing counter is kept for X.25 circuits:

Corruption loss

The following Data Link counters are kept for a permanent X.25 circuit:

Bytes received
Bytes sent
Data blocks received
Data blocks sent

Locally initiated resets
Remotely initiated resets
Network initiated resets

The following Data Link counters are kept for Ethernet circuits:

Bytes received
Bytes sent
Data blocks received
Data blocks sent
User buffer unavailable

3.5 Lines

Lines are the lowest level communications path. Lines provide physical communications. Lines are the media over which circuits operate.

A line is identified individually with a line name. The Data Link layer contains the master list of line names and ensures their uniqueness. A line name is an id-string.

Group line identifications are as follows:

ACTIVE LINES - All lines that are in the ON or SERVICE state.

KNOWN LINES - All lines that have a name.

SIGNIFICANT LINES - All known lines that have at least one parameter.

Many line parameters, counters, and events depend on the communications protocol being used for a particular line. There are currently three protocols: DDCMP, LAPB, and Ethernet. DDCMP is for DECnet communications. LAPB is for DECnet and/or X.25 communications. Ethernet is for DECnet or Ethernet communications.

3.5.1 Line Parameters

There are five groups of line parameters:

1. Common line parameters (Section 3.5.1.1)

2. Non-Ethernet line parameters (Section 3.5.1.2)

3. DDCMP line parameters (Section 3.5.1.3)

4. LAPB line parameters (Section 3.5.1.4)

5. Ethernet line parameters (Section 3.5.1.5)

3.5.1.1 Common Line Parameters

The following parameters are common to all lines:

COUNTER TIMER seconds

This value represents the Network Management timer whose expiration causes a line counter logging event. The counters logged depend on the line protocol and are described elsewhere. The line counters are recorded as data in a logging event and then zeroed. If no counter timer value is set, the line's counters are not automatically logged. Seconds is a decimal integer in the range 1-65535.

DEVICE device-specification

This value represents the Physical Link device to be used on the line. A device-specification contains the following:

dev A device mnemonic (Table 10, Section 7.4)

c A controller number

u A unit number

These fields represent the actual local hardware for the device. If the device is not a multiple line controller, the unit number is not allowed. The device-specification is an id-string in the following format:

dev-c

or

dev-c-u

PROTOCOL protocol-name

This value represents the Data Link protocol to be used on the line. The protocol-name values are as follows:

DDCMP CONTROL

This line is the control station for a DDCMP multipoint group. It can be the line for multiple circuits, each of which has a unique physical tributary address.

DDCMP DMC

This line is in DMC emulator node.

DDCMP POINT

This line is one end of a point-to-point DDCMP connection. It can be the line for only one circuit.

DDCMP TRIBUTARY

This line is a tributary end of a DDCMP multipoint group. It can be the line for only one circuit.

LAPB

This line uses the X.25 level 2 protocol and can be a line for the X25-PROTOCOL module.

ETHERNET

This line uses the Ethernet protocol for the Ethernet.

RECEIVE BUFFERS number

This value represents the number of receive buffers reserved for the line. It is a decimal number in the range 1-65535.

STATE line-state

This value represents Network Management operational state as described in the state and substate model in Section 3.6.

Substate

This value represents the line's read-only Network Management substate as described in Section 3.6.

3.5.1.2 Non-Ethernet Line Parameters

The following parameters are common to all non-Ethernet lines:

CLOCK clock-mode

This value represents the Physical Link hardware clock mode for the line device. The values for clock-mode are:

INTERNAL For software controllable loopback use of the clock. On those devices that can support this mode, it causes the device to supply a clock signal such that all transmitted messages can be looped back from outside the device. This may require manual intervention other than the setting of this parameter value. For example, the operator may have to connect a loopback plug in place of the normal line.

EXTERNAL For normal clock operating mode, where the clock signal is supplied externally to the controller.

CONTROLLER controller-mode

This value represents the Physical Link hardware controller mode for the line device. The values for controller-mode are:

LOOPBACK For software controllable loopback of the controller. On those devices that can support this mode, it causes all transmitted messages to be looped back from within the controller itself. This is accomplished without any manual intervention other than the setting of this parameter value.

NORMAL For normal controller operating mode.

DUPLEX duplex-mode

This value represents the Physical Link hardware duplex mode of the line device. The possible modes are:

FULL Full-duplex

HALF Half-duplex

RETRANSMIT TIMER milliseconds

This value represents the amount of time before the Data Link retransmits a block on the line. On half-duplex lines, this parameter is the select timer. Milliseconds is a decimal integer in the range 1-65535. If not set, the default is 3000 (3 seconds).

3.5.1.3 DDCMP Line Parameters

DDCMP lines support the Network Management service functions of dump, load, and active and passive line loopback. The following parameter applies to DDCMP lines:

SERVICE TIMER milliseconds

This value represents the amount of time allowed to elapse before a Data Link receive request completes while doing service operations. Milliseconds is a decimal integer in the range 1-65535.

The following parameters apply to DDCMP CONTROL lines:

DEAD TIMER milliseconds

This value represents the number of milliseconds between polls of one of the set of dead tributaries. Milliseconds is a decimal integer in the range 1-65535. If not set, the default is 10000

(10 seconds).

DELAY TIMER milliseconds

This value represents the minimum number of milliseconds to delay between polls. The delay timer limits the effect of a very fast control station on slow tributaries. Milliseconds is a decimal integer in the range 1-65535. If not set, there is no delay.

SCHEDULING TIMER milliseconds

This value represents the number of milliseconds between recalculation of tributary polling priorities. Milliseconds is a decimal integer in the range 50-65535. If not set, the default is 200.

STREAM TIMER milliseconds

This value represents the number of milliseconds a tributary or a half duplex remote station is allowed to hold the line. Milliseconds is a decimal integer in the range 0-65535. If not set, the default is 6000 (6 seconds).

NOTE

This parameter can also be applied to half-duplex lines of type DDCMP POINT.

3.5.1.4 LAPB Line Parameters

LAPB lines support the Network Management service function of active loop. The following parameters apply to lines using the LAPB protocol:

HOLDBACK TIMER milliseconds

This parameter defines the length of time a Data Link acknowledgment can be held back to wait for a chance to piggy-back on a data message. If no value is set, no holdback is allowed. Milliseconds is a decimal integer in the range 1-65535.

MAXIMUM BLOCK byte-count

This value represents the Data Link maximum block size on the line. Byte-count is a decimal integer in the range 1-65535.

MAXIMUM RETRANSMITS block-count

This value represents the maximum number of Data Link retransmits of a block on the line. If no value is set, there is no maximum. Block-count is a decimal integer in the range 1-255.

MAXIMUM WINDOW block-count

This value represents the Data Link maximum number of unacknowledged transmitted blocks on the line. Block count is a decimal integer in the range 1-255.

SERVICE service-control

This value indicates whether or not Network Management service operations (loading, dumping, loopback testing) are allowed for the line. The service-control values are as follows:

ENABLED SERVICE state and/or service functions are allowed.

DISABLED SERVICE state and/or service functions are not allowed.

3.5.1.5 Ethernet Line Parameters

Ethernet lines do not support any Network Management service functions. The following parameters apply to lines using the Ethernet protocol:

HARDWARE ADDRESS ethernet-address

This read-only parameter is the Ethernet address associated with the line device hardware.

3.5.2 Line Counters

Network management displays or zeroes line counters pertaining to a particular line as a group. Some line counters are specific to DDCMP lines, some to LAPB lines, and some to Ethernet lines. The following Network Management counter is kept for lines:

Seconds since last zeroed

The following Data Link counters are kept for DDCMP lines:

Remote station errors
Local station errors

The following Data Link counters are kept for an LAPB line:

Bytes received
Bytes sent
Data blocks received
Data blocks sent
Data errors inbound
Data errors outbound
Remote reply timeouts
Local reply timeouts

Remote buffer errors
Local buffer errors
Remote process errors
Local process errors

The following Data Link counters are kept for an Ethernet line:

Bytes received
Bytes sent
Data blocks received
Data blocks sent
Multicast bytes received
Multicast data blocks received
Data blocks sent, initially deferred
Data blocks sent, single collision
Data blocks sent, multiple collisions
Send failure
Collision detect check failure
Receive failure
Unrecognized frame destination
Data overrun
System buffer unavailable
User buffer unavailable

3.6 Circuit and Line State and Substate Model

This section describes the Network Management state and substate model. This model applies to both circuits and lines, referred to generically as links. There is one model for the relationships between the states and substates. This model is applied independently to both circuits and lines. In other words, all of the states and substates that apply to circuits apply equally and independently to lines. Note that this is an architectural model and the functions and states that can be applied to a particular circuit or line will vary depending upon the Data Link protocol and the actual implementation.

In the following discussion of the model, the term link is used to include either circuit or line. This Network Management function is called Data Link Service to include both circuits and lines.

There are three internal state machines that are of interest, one each for the high level user's internal view of a link, the Data Link protocol's exhibited view of a link, and the Network Management Data Link Service view of a link. These state machines are first presented independently. They are then related to one another through the externally visible states and substates.

The purpose of these state machines is to define the Network Management abstractions for the operation of links. These abstractions can represent any high level user or low level Data Link, within the bounds of the functions actually provided by that module. A mapping of the Network Management link state machines to actual internal states of other architectural components is in Appendix C.

Operation of various algorithms on circuits and lines differs according to how maintenance traffic is handled within their data link protocol. In some data link protocols (i.e. DDCMP and LAPB) maintenance traffic is exclusive of normal traffic and vice versa. The link is either in normal mode or maintenance mode. In others (i.e. Ethernet) maintenance traffic is completely independent of normal traffic and thus can occur concurrently. Whenever this makes a difference in the remainder of this specification, these will be referred to as exclusive maintenance or concurrent maintenance links, respectively.

3.6.1 High Level Link User States

In the case of a circuit, the high level user is either the circuit owner for a circuit that has an owner or the current user for an opened circuit that does not have an owner. For a line, the high level user is the Data Link protocol module. The state machine presented here models the Network Management view of the high level user. The user's internal operation may actually be different, but Network Management must be able to view it essentially according to this model.

The high level user internally considers the link as being in one of four states:

1. Off -- the link is not to be used.

2. Start - the link is to be or is being initialized. This includes both the low level going to run state and any initialization needed by the high level.

3. Fail -- the link startup process failed permanently. This state may not exist in some high level users.

4. Run -- the link is in normal running state.

Figure 3 shows the states and the allowed transitions.

```
     .---------.              .----------.
     |         |<-------------|          |
     |   OFF   |              |   RUN    |
     |         |              |          |
     `---------'<--------.    `----------'
          A      \         \     ||   /\
          |       \         \    ||   ||
          |        \         \   ||   ||
          |         \         \  ||   ||
          |          \         \ \/   ''
     .---------.      `------->.----------.
     |         |               |          |
     |  FAIL   |.------------->|  START   |
     |         |<============='|          |
     `---------'               `----------'
```

```
---->  Network Management Command

====>  Protocol Operation
```

Figure 3. High Level Link User State Diagram

Table 1 defines the state transitions and the events that cause them. Network Management commands are represented by the phrase "SET STATE state" where state is the controllable state. A hyphen indicates no change of state, N/A indicates Not Allowed (impossible).

Table 1
High Level Link State Transitions and Their Causes

Event	Old State	Off	Start	Fail	Run
SET STATE OFF		-	Off	Off	Off
SET STATE ON		Start	-	Start	-
SET STATE SERVICE		Off	Off	Off	Off
Startup successful		N/A	Run	N/A	N/A
Startup failed		N/A	*Fail	N/A	N/A
Lower level left run state		N/A	-	N/A	Start

* This transition takes place according to the user's algorithms or not at all if not supported by the user.

3.6.2 Data Link States

The Data Link states are those exhibited for any higher level user. The actual internal states may differ according to the particular Data Link protocol, but Network Management and the high level user must be able to perceive them as described here.

Data Link exhibits the link as being in one of four states. In this context, Data Link means a lower level perception of the link, since in the case of a line the high level user is actually part of Data Link. the low level Data Link states are:

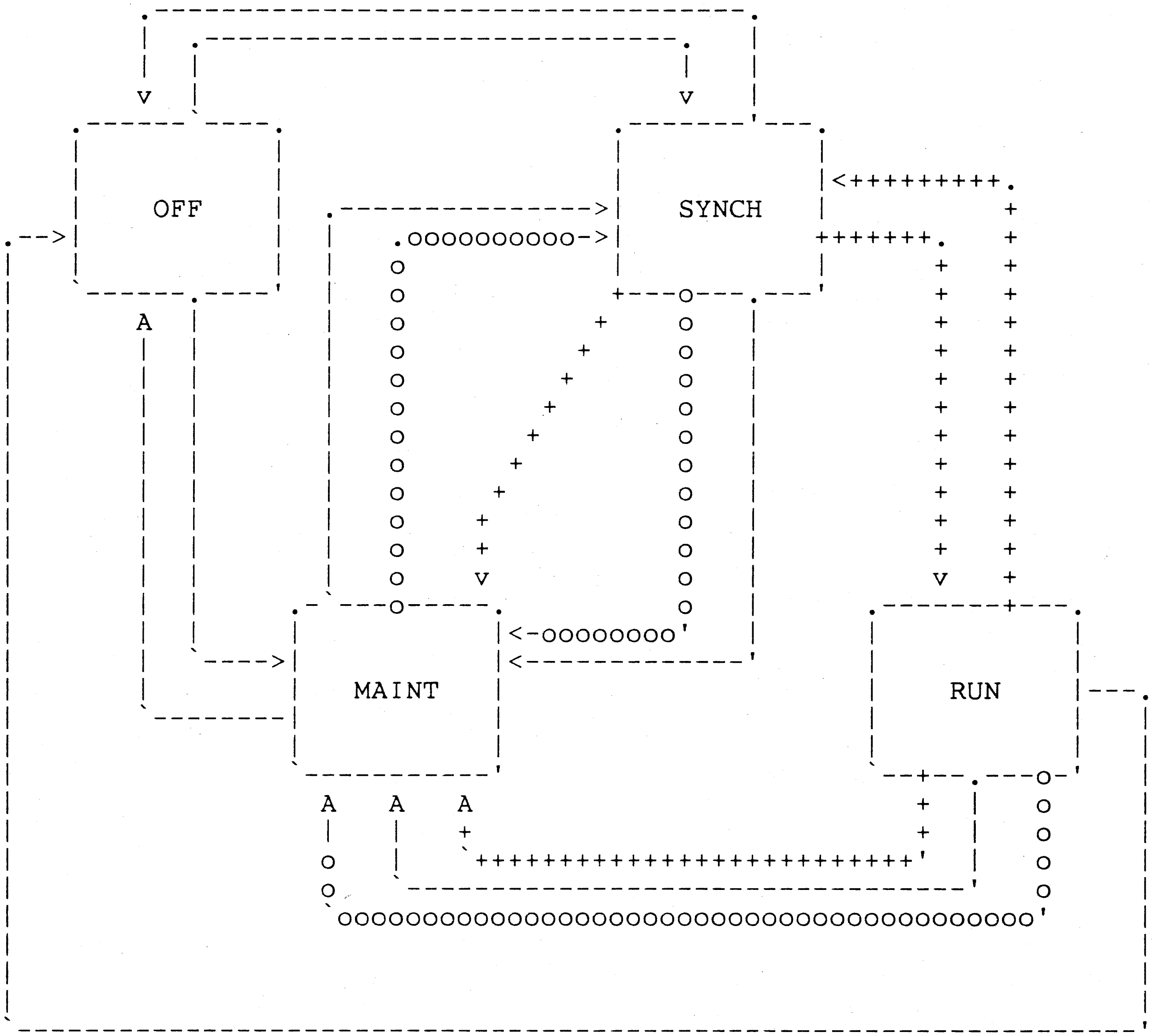
1. Off -- the link is not to be used.

2. Synch -- the link is to be or is being initialized.

3. Maint -- the link is to be used for maintenance purposes. This state only applies to exclusive maintenance links.

4. Run -- the link is in normal running state.

Figure 4 shows the states and the allowed transitions.




```
--->  Network Management Command

oo->  Network Management Operation

+++>  Protocol Operation
```

Figure 4. Data Link State Diagram

Table 2 defines the Data Link state transitions and the events that cause them. Network Management commands are represented by the phrase "SET STATE state" where state is the controllable state. A hyphen indicates no change of state, N/A indicates Not Allowed (impossible).

Table 2
Data Link State Transitions and Their Causes

Event Old State	Off	Synch	Maint	Run
SET STATE OFF	-	Off	Off	Off
SET STATE ON	Synch	-	Synch	-
SET STATE SERVICE	Maint	Maint	-	Maint
*Set state ON-AUTOSERVICE	N/A	Maint	N/A	Maint
*Reset ON-AUTOSERVICE state	N/A	N/A	Synch	N/A
Data Link Service open	N/A	Maint	-	Maint
Data Link Service close	N/A	N/A	**Synch	N/A
Maintenance message received	N/A	Maint	-	Maint
Synchronization successful	N/A	Run	N/A	N/A
Synchronization lost	N/A	N/A	N/A	Synch

* Controllable states for Link Watcher only.

** If controllable state in ON, otherwise no change.

3.6.3 Network Management Data Link Service States

Network Management Data Link Service is the arbiter of link states, maintaining the proper relationships as required by Network Management. Data Link service maintains its own internal states for a link according to the following description.

This discussion is directed only at exclusive maintenance links. Concurrent maintenance links are either on or off, regardless of whether they carry normal or maintenance traffic. Furthermore, all their maintenance functions are handled by the link maintenance modules, which are dedicated to the processing of maintenance functions on the links that they own.

Network Management Data Link Service perceives the link as being in one of seven states:

1. Off -- the link is not to be used.

2. Passive -- the link is in use by its user and is being monitored by Network Management Data Link Service.

3. Passive open -- the link is temporarily in use for some Network Management operation such as down-line load and is to return to its user when done.

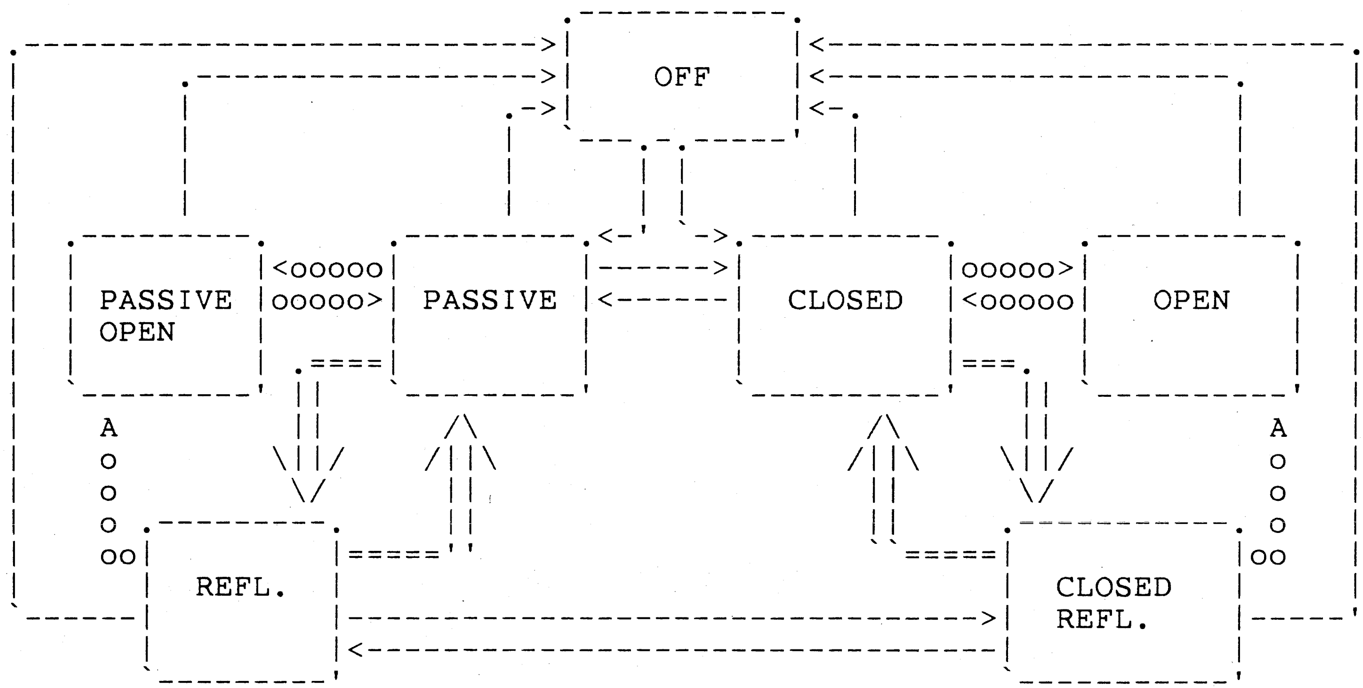
4. Reflecting -- the link is reflecting loopback messages and is to return to its user when done.

5. Closed -- the link is reserved for some Network Management operation.

6. Open -- the link is open for some Network Management operation.

7. Closed reflecting -- the link is reflecting loopback messages but is reserved for some Network Management operation on demand.

Figure 5 shows the states and the allowed transitions.



---> Network Management Command

ooo> Network Management Operation

===> Protocol Operation

Figure 5. Network Management Data Link Service State Diagram

Table 3 defines the state transitions and the events that cause them. Network Management commands are represented by the phrase "SET STATE state" where state is the controllable state. A hyphen indicates no change of state, N/A indicates Not Allowed (i.e., impossible).

Table 3
Data Link Service State Transitions and Their Causes

Event	Off	Pass	Pass Open	Refl	Clos	Open	Clos Refl
SET STATE OFF	-	Off	Off	Off	Off	Off	Off
SET STATE ON	Pass	-	-	-	Pass	Pass	Refl

SET STATE SERVICE	Clos	Clos	Clos	Clos Refl	-	-	-
Data Link Service open	N/A	Pass Open	N/A	Pass Open	Open	N/A	Open
Data Link Service close	N/A	N/A	Pass	N/A	N/A	Clos	N/A
Loop message received	N/A	Refl	N/A	-	Clos Refl	N/A	-
Communication failure or non-maint message received	N/A	-	-	Pass	N/A	-	Clos

3.6.4 Controllable and Observable States and Substates

This section shows the relationships between the three previously defined state machines, and relates them to the states and substates that can be controlled and observed through Network Management.

There are four link states that can be controlled through Network Management.

1. CLEARED -- Some space is reserved for the link, but no other data bases or parameters for the link are present. The link cannot be used in any way. If they exist, all three state machines are in the off state.

2. OFF -- The link data bases and parameters are present, but the link is not to be used by any network or network-related software. The link is functionally non-existent. All three state machines are in the off state.

3. ON -- The link is available to a high level user for normal use, with the exception of temporary overrides for service functions. The high level user is in any state but off. Data Link is in any state but off. Data Link Service is in passive, passive open, or reflecting state.

4. SERVICE -- The link is reserved for the active service functions: load, dump, and loop. The link can provide passive loop if no active service function is in progress. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the closed, open, or closed reflecting state. This state does not apply to concurrent maintenance links.

There is one additional state that can be requested by the Link Watcher:

ON-AUTOSERVICE -- The link is temporarily reserved for the active service functions. It is to be returned to its high level user when the Link Watcher is done. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the closed state. This state does not apply to concurrent maintenance links.

There are fourteen substates that can be observed through Network Management. These substates apply to the ON and SERVICE states unless otherwise noted. OFF and CLEARED do not have substates. The only substates that apply to concurrent maintenance links are running and FAILED.

1. Running -- The link is in normal use by its high level user. This is the default substate of ON. The high level user and Data Link are in the run state. Data Link Service is in the passive state.

2. Idle -- The link is not being used for anything. This is the default substate of SERVICE. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the closed state.

3. SYNCHRONIZING -- The link is engaged in low level Data Link synchronization. This is a substate of ON. The high level user is in the start state. Data Link is in any state except run or off. Data Link Service is in the passive state.

4. STARTING -- The link is synchronized and is in its high level user's startup cycle. This is a substate of ON. The high level user is in the start state. Data Link is in the run state. Data Link Service is in the passive state.

5. FAILED -- The link permanently failed its high level user's startup cycle. This is a substate of ON. The high level user is in the fail state. Data Link is in any state. Data Link Service is in the passive state.

6. REFLECTING -- The link is engaged in passive loopback. The high level user is in the off or start state. Data Link is in the maint state. Data Link Service is in the reflecting or closed reflecting state.

7. LOADING -- The link is engaged in down-line load. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the open state.

8. DUMPING -- The link is engaged in up-line dump. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the open state.

9. LOOPING -- The link is engaged in active loopback. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the open state.

10. TRIGGERING -- The link is engaged in a down-line trigger. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the open state.

11. AUTOSERVICE -- The link is reserved for Link Watcher use. This appears as a substate of ON. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the closed state.

12. AUTOLOADING -- The link is engaged in down-line load for the Link Watcher. This appears as a substate of ON. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the open state.

13. AUTODUMPING -- The link is engaged in up-line dump for the Link Watcher. This appears as a substate of ON. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the open state.

14. AUTOTRIGGERING -- The link is engaged in a down-line trigger for the Link Watcher. This appears as a substate of ON. The high level user is in the off state. Data Link is in the maint state. Data Link Service is in the open state.

3.7 Modules

Modules are components that do not fit into the other entity classifications. Module identification is a module name. The Network Management layer contains the master list of module names and ensures their uniqueness. A module name is an id-string. Since module names are predefined by Network Management, they can be abbreviated according to the same rules applied to keywords (Section 4.2.4).

3.7.1 X.25 Access Module

The name of the X.25 access module is X25-ACCESS.

The access module data base contains the information necessary to connect to the X.25 Server for one or more networks. This information is indexed by network name. Functions that reference this data base must indicate to which network name they apply, except for the case where only one network name is defined.

The user can add and remove network names and parameters. The user can also modify parameters for a network name or names. Network Management can display information about the X.25 access module by network name or by all known network names.

The network name parameter formats are as follows:

KNOWN NETWORKS

All of the network names known to the access module.

NETWORK network-name

The name of a specific network for the access module. A network name is an id-string.

Each network name is associated with a node identification and, optionally, access control information. The parameters kept by network name are as follows:

ACCOUNT account

This is the access control account field value. The access routines use this value when connecting to the server. If no account is set, none is included in the access control on connect by the access module. Account is a string of one to 39 characters.

NODE node-id

The identification of the node to be used by the access routines in connecting to a server. Node-id is a standard Network Management node identification (Section 3.1).

PASSWORD password

The access control password field value to be used by the access routines in connecting to the server. If no password is set, none is included in the access control on connect by the access module. Password is a string of one to 39 characters.

USER user

The access control user field value to be used by the access routines in connecting to the server. If no user is set, none is included in the access control on connect by the access module. User is a string of one to 39 characters.

3.7.2 X.25 Protocol Module

The name of the X.25 protocol module is X25-PROTOCOL.

The protocol module data base contains the information necessary to maintain switched and permanent virtual circuits through a public data network over its assigned X.25 lines. Most of this information is indexed by the local DTE addresses. Functions that reference this data base must indicate to which DTE address they apply, except for the case where only one local DTE address is defined.

The user can add and remove local DTE addresses and parameters. The user can also modify parameters for a local DTE address or addresses. Network Management can display information from the X.25 protocol module data base by local DTE address or for all known local DTE addresses.

Closed user group information is indexed by group name. Functions that reference this part of the protocol module data base must indicate to which group they apply. Groups can be added and removed, along with their parameters. Information can be requested by group name or for all known groups.

There are also protocol module counters and parameters that are independent of local DTE addresses.

3.7.2.1 X.25 Protocol Module Parameters

The local DTE address independent protocol module parameters are as follows:

CALL TIMER seconds

> This value indicates the maximum elapsed seconds before the X.25 protocol module will send a clear on outgoing calls from the local DTE for which no response has been received. If no timer is set, there is no clear sent. Seconds is a decimal integer in the range 1-255.

CLEAR TIMER seconds

> This is the retransmit timer for outgoing clear packets from the local DTE. If no timer is set, there is no retransmission. Seconds is a decimal integer in the range 1-255.

DEFAULT DATA byte-count

> This parameter is the default data size for switched circuits. Byte-count is a decimal integer in the range 1-65535.

DEFAULT WINDOW block-count

> This parameter is the default number of unacknowledged transmitted blocks on a switched circuit. Block-count is a decimal integer in the range 1-255.

MAXIMUM DATA byte-count

> This parameter is the maximum data size for all circuits. Byte-count is a decimal integer in the range 1-65535.

MAXIMUM CLEARS retry-count

> This value is the maximum number of times that the X.25 protocol handler is to retry the sending of a clear for switched circuits.

If no value is set, there is no maximum. Retry-count is a decimal integer in the range 1-255.

MAXIMUM RESETS retry-count

This value is the maximum number of times that the X.25 protocol handler is to retry the sending of a reset. If no maximum is set, there is no maximum. Retry-count is a decimal integer in the range 1-255.

MAXIMUM RESTARTS retry-count

This value is the maximum number of times that the X.25 protocol handler is to retry the sending of a restart. If no maximum is set, there is no maximum. Retry-count is a decimal integer in the range 1-255.

MAXIMUM WINDOW block-count

This value is the maximum number of unacknowledged transmitted blocks on a switched circuit. Block count is a decimal integer in the range 1-255.

NETWORK network-name

This network name value can be used by the X.25 protocol handler to determine network specific characteristics and values. It is also used as the network name for outgoing and incoming calls.

The network-name is an id string.

RESET TIMER seconds

This parameter is the retransmit timer for outgoing reset packets from the local DTE. If no timer is set, there is no retransmission. Seconds is a decimal integer in the range 1-255.

RESTART TIMER seconds

This parameter is the retransmit timer for outgoing restart packets from the local DTE. If no timer is set, there is no retransmission. Seconds is a decimal integer in the range 1-255.

To access the local DTE indexed parameters, a local DTE address must be specified or assumed. A local DTE address can only be assumed if only one is known. The local DTE address parameter is as follows:

KNOWN DTES

This parameter refers to all of the local DTE addresses known to the protocol module.

DTE dte-address

This parameter represents a particular local DTE address for the protocol module. A local DTE address is a decimal integer of one to sixteen digits.

Each local DTE address is to be associated with the information needed to process the virtual circuits associated with that local DTE.

The parameters kept by local DTE address are as follows:

ACTIVE CHANNELS count

This read-only value is the number of switched virtual circuit logical channel numbers that are currently in use. These are the channels specifically defined with the channels parameter (see below). Active channels include those allocated from the channel list for either outgoing or incoming switched virtual circuits.

ACTIVE SWITCHED circuit-count

This read-only value is the number of currently active switched circuits. This is the total number of switched virtual circuits active. It includes both those that were included in the active channels count plus any incoming calls that did not use one of the channels in the channel list.

CHANNELS list

This parameter is the list of logical channel numbers that can be used for outgoing calls or possibly taken by incoming calls.

List is one or more logical channel numbers. Multiple channel numbers are separated with hyphens to indicate ranges and commas to indicate individual numbers. The order of the numbers in the list defines the order in which the logical channel numbers are to be allocated by the protocol module.

For example, the command

```
SET MODULE X25-PROTOCOL CHANNELS 20-10,8,3
```

Sets 20 as the first channel number to use, counting down from there to 10, then 8, and finally 3.

COUNTER TIMER seconds

This parameter is the Network Management timer whose expiration causes a module counter logging event. The module counters are recorded as data in a logging event and then zeroed. If no counter timer is set, the module counters are not automatically logged.

Seconds is a decimal integer in the range 1-65535.

LINE line-id

This value represents a LAPB line to be used by the X.25 protocol module.

MAXIMUM CHANNELS count

This value is the read-only number of logical channels defined.

MAXIMUM CIRCUITS count

This is the parameter that indicates the maximum number of circuits that the DTE can open at one time. This includes both outgoing and incoming calls. Incoming calls can be given logical channel numbers outside the given channels list. The count is a decimal integer in the range 1-65535. Default is 255.

STATE dte-state

This value represents the operational state of a local DTE. The possible states are as follows:

ON

The DTE is allowed to operate normally.

OFF

The local DTE is not allowed to operate at all. Any existing virtual circuits are terminated immediately.

SHUT

The local DTE will not allow any new virtual circuits to be formed. Existing virtual circuits are undisturbed. When the final existing virtual circuit terminates, the state automatically goes to OFF.

SUBSTATE dte-substate

There are a number of substates a DTE can be in while it is in ON state. The substate depicts the link status between the DTE and the DCE.

The possible substates are:

UNSYNC

The link to the DCE not in the "RUNNING" state.

SYNC

The packet level has initiated a RESTART for the DTE.

RUNNING

Normal operation.

The group name parameter is as follows:

KNOWN GROUPS

> This represents all of the closed user groups known to the protocol module.

GROUP group-name

> This indicates a particular closed user group for the protocol module. A group-name is an id string.

Each group name is to be associated with the information needed to use the group through the X.25 network.

The parameters kept by group name are as follows:

DTE dte-address

> This value represents the local DTE address to which the group number belongs. When setting this value, it must be accompanied by a group number parameter. Dte-address is a decimal integer of 1 to 16 digits.

NUMBER group-number

> This is the closed user group number. When setting this value, it must be accompanied by a dte address parameter. Group-number is a decimal integer in the range 0-9999.

TYPE group-type

> This is the closed user group type. The only group type defined is BILATERAL. If no type is set, the group is not bilateral.

3.7.2.2 X.25 Protocol Module Counters

These counters are:

- Seconds since last zeroed
- Bytes received
- Bytes sent
- Data blocks received
- Data blocks sent
- Calls received
- Calls sent
- Fast selects received
- Fast selects sent
- Maximum switched circuits active
- Maximum channels active
- Received call resource errors
- Locally initiated resets
- Remotely initiated resets

Network initiated resets
Restarts

3.7.3 X.25 Server Module

The name of the X.25 Server module is X25-SERVER.

The server module data base contains the information necessary to map incoming X.25 calls to a DECnet process. This information is indexed by destination name. Functions that reference this data base must indicate to which destination name they apply, except for the case that there is only one destination name defined.

Destination names can be added and removed, along with their parameters. Parameters for a destination name or names can be modified. Information can be requested by destination name or for all known destination names.

There are also server counters and some server parameters that are independent of destination names.

3.7.3.1 X.25 Server Module Parameters

The destination name independent server parameters are as follows:

ACTIVE CIRCUITS count

> This is the read-only module parameter that indicates the number of circuits that the module currently has open.

COUNTER TIMER seconds

> This is the Network Management timer whose expiration causes a module counter logging event. The module counters are recorded as data in a logging event and then zeroed. If no counter timer is set, the module counters are not automatically logged. Seconds is a decimal integer in the range 1-65535.

MAXIMUM CIRCUITS count

> This is the module parameter that indicates the maximum number of circuits that the module can have open at one time. Count is a decimal integer in the range 1-65535.

The destination name parameter is as follows:

KNOWN DESTINATIONS

> This refers to all of the destination names known to the server module.

DESTINATION destination-name

This indicates the name of a specific destination for the server module. A destination name is an id-string.

Each destination name is to be associated with a DECnet node and object identification and with the necessary X.25 related information to recognize the incoming call. The algorithms for incoming call recognition are in the X.25 Gateway Access specification.

The parameters kept by destination name are as follows:

ACCOUNT account

This is the access control account field value to be used in connecting to the DECnet destination of an incoming call. If no account value is set, the server will not use one in the connect. Account is a string of 1 to 39 characters.

CALL MASK hex-value

This is the call mask value to be used to identify the destination for an incoming call. If no call mask is set, none will be used in the identification process. Hex-value is a hexadecimal number of 1 to 32 digits.

CALL VALUE hex-value

This is the call data value to be used to identify the destination for an incoming call. If no call value is set, none will be used in the identification process. Hex-value is a hexadecimal number of 1 to 32 digits.

GROUP group-name

This is the closed user group name to be used to identify the destination for an incoming call. If no group name is set, none will be used in the identification process. The group-name value is an id string.

NODE node-id

This is the identification of the node to be used in connecting to the DECnet destination of an incoming call. Node-id is a standard Network Management node identification.

NUMBER call-number

This is the full remote DTE address to be used to identify the destination for an incoming call. If no remote DTE address is set, none will be used in the identification process. Call-number is a string of 1 to 16 numeric digits and/or asterisks (*).

OBJECT object-id

This is the object identification to be used in connecting to the DECnet destination of an incoming call. Object-id is either a string of 1 to 16 characters for named objects or a decimal number in the range 1-255 for numbered objects. If a named object has a name that looks like a number for a numbered object, the name is specified on input in quotes to indicate that the object-id is a string rather than a number. On output there is no differentiation.

PASSWORD password

This is the access control password field value to be used in connecting to the DECnet destination of an incoming call. If no password value is set, the server will not use one in the connect. Password is a string of 1 to 39 characters.

PRIORITY priority

This is the priority with which the X.25 set of information is to be used. The highest priority is 255 and the lowest is 0. Priority is a decimal integer in the range 0-255.

SUBADDRESSES range

This is the range of local DTE subaddresses to be used to identify the destination for an incoming call. If no subaddresses are set, none are used in the identification process. The range value consists of one or two subaddresses. A subaddress is a decimal integer in the range 0-9999. If two subaddresses are provided, specifying a range, they are separated by only a single hyphen and the second must be greater than the first.

USER user

This is the access control user field value to be used in connecting to the DECnet destination of an incoming call. If no user value is set, the server will not use one in the connect. User is a string of 1 to 39 characters.

3.7.3.2 X.25 Server Module Counters

These counters are:

Seconds since last zeroed
Maximum circuits active
Incoming calls rejected, no resources
Logical links rejected, no resources

3.7.4 Link Maintenance Modules

The names of the link management modules are LOOPER, LOADER, CONSOLE, and CONFIGURATOR.

The link maintenance modules provide the network manager with entities that can own Ethernet circuits for link service functions such as loop testing and down-line load. Some of the link maintenance modules have parameters for controlling or observing their operation.

The looper, loader, and console modules are the only link maintenance modules that can be CIRCUIT owners. They each own one circuit on every Ethernet line that is to have their related service functions.

The configurator module is a user of the services represented by the console module. The configurator module can provide information from a single console request for system identification or can listen through the console and construct a list of the systems on an Ethernet line.

3.7.4.1 Console Module Parameter

The console module parameter is as follows:

RESERVATION TIMER seconds

> This value indicates the number of seconds that the console will stay reserved without hearing from the system that reserved it. Seconds is a decimal integer in the range 1-65535.

3.7.4.2 Loader Module Parameter

The loader module parameter is as follows:

ASSISTANCE control

> This value indicates whether or not this node will respond to the dump/load assistance multicast address. The control values are as follows:
>
> ENABLED The node will respond.
>
> DISABLED The node will not respond.

3.7.4.3 Looper Module Parameter

The looper module parameter is as follows:

ASSISTANCE control

This value indicates whether or not this node will respond to the loopback assistance multicast address. The control values are as follows:

ENABLED The node will respond.

DISABLED The node will not respond.

3.7.4.4 Configurator Module Parameters

All configurator module parameters are qualified with the circuit identification of the Ethernet circuit to which they relate. The circuit identification parameter is as follows:

CIRCUIT circuit-id

This indicates the circuit of interest. It must be an Ethernet circuit. Circuit-id is an id-string.

The parameters kept by circuit identification are as follows:

ELAPSED TIME hours:minutes:seconds

This read-only value is the amount of time that surveillance has been enabled on the channel. Hours is a decimal integer in the range 0-65535, minutes and seconds are decimal integers in the range 0-59.

SURVEILLANCE control

This value indicates whether or not a list of active systems is to be kept for the circuit. The control values are as follows:

ENABLED The list is kept.

DISABLED The list is not kept.

The default value is DISABLED.

Within a circuit, many parameters are further qualified by the remote system's physical address. The physical address parameter is as follows:

PHYSICAL ADDRESS ethernet-address

This read-only value is the Ethernet address of a remote system on the circuit. When used as a qualifier on a display request, it causes an active request to the console at the specified address and returns the resulting information.

The parameters qualified by physical address are as follows:

COMMAND SIZE bytes

This read-only value is the number of bytes in the remote system's console carrier protocol command buffer. Bytes is a decimal number in the range 1-65535.

CONSOLE USER ethernet-address

This read-only value is the Ethernet address of the system that has the remote system's console reserved. It is either an Ethernet address or the word "NONE".

DATA LINK data-link-type

This read-only value is the type of data link protocol being used on the circuit over which the remote system is communicating. Its values are defined in the DNA Low Level Maintenance Operation specification.

DATA LINK BUFFER SIZE data-link-type

This read-only value is the size of data link buffer being used on the circuit over which the remote system is communicating. Its values are defined in the DNA Low Level Maintenance Operation specification.

DEVICE device-type

This read-only value is the type of device over which the remote system is communicating on the circuit. It is one of the standard line devices.

FUNCTIONS function-list

This read-only value is the list of maintenance functions that the remote system supports. The list of items is one or more of the following:

BOOT	Remote controlled boot
CARRIER	Console carrier protocol
COUNTERS	Data link counter read
DUMP	Up-line dump
LOAD	Multi-block down-line load
LOOP	Loopback
PRIMARY	Primary loader

HARDWARE ADDRESS ethernet-address

This read-only value is the Ethernet address that is attached to the remote system hardware. It may be relative to the particular device through which the remote system is communicating on the circuit.

LAST REPORT day-month hour:minute:second

This read-only value is the date and time of the last time the remote system reported in on a circuit that is under surveillance. Day is a decimal integer in the range 1-31. Month is the name of the month. Hour is a decimal integer in the range 0-23. Minute and second are decimal integers in the range 0-59.

MAINTENANCE VERSION n.n.n

This read-only value is the maintenance protocol version of the remote system, consisting of the version number, the Engineering Change Order (ECO) number, and the user ECO number (for example, 3.0.0).

RESERVATION TIMER seconds

This read-only value is the maximum time that the remote system's console will remain reserved without a message from the console user. Seconds is a decimal integer in the range 1-65535.

RESPONSE SIZE bytes

This read-only value is the number of bytes in the remote system's console carrier protocol response buffer. Bytes is a decimal number in the range 1-65535.

SYSTEM PROCESSOR processor-type

This read-only value is the type of main processor on the remote system. Its values are defined in the DNA Low Level Maintenance Operation specification.

SOFTWARE IDENTIFICATION software-id

This read-only value identifies the software that the remote system is supposed to be running. It is defined the same as NODE SOFTWARE IDENTIFICATION.

3.8 Events

Events are significant occurrences in the DNA layers that the Network Management Event Logger records. This section lists the events recorded according to the entity and layer with which they are associated. Section 3.9 describes the event parameters. Section 5.5 describes the operation of the Event Logger. Section 6.13 specifies the Event message binary data format. Section 7.12 specifies the events. Section 7.13 specifies the binary formats and values for the event parameters.

3.8.1 Events Not Related to an Entity

The Event Logger records the following Network Management event.

Event records lost

The Event Logger records the following Session Control events:

Local node state change
Access control reject

The Event Logger records the following End Communication events:

Invalid message
Invalid flow control

3.8.2 Node Events

The Event Logger records the following Network Management node events:

Automatic counters
Counters zeroed

The Event Logger records the following End Communication node event:

Data base reused

The Event Logger records the following Routing node event:

Node reachability change

3.8.3 Circuit Events

The Event Logger records the following Network Management circuit events:

Automatic counters
Automatic service
Counters zeroed
Passive loopback
Aborted service request

The Event Logger records the following Routing circuit events:

Node unreachable packet loss
Node out-of-range packet loss
Oversized packet loss
Packet format error
Partial routing update loss
Verification reject
Circuit down, circuit fault
Circuit down
Circuit down, operator initiated
Adjacency down
Adjacency down, operator initiated
Circuit up
Adjacency up

Initialization failure, circuit fault
Initialization failure
Initialization failure, operator initiated
Area reachability change
Adjacency rejected

The Event Logger records the following events for DDCMP circuits:

Locally initiated state change
Remotely initiated state change
Protocol restart received in maintenance mode
Send error threshold
Receive error threshold
Select error threshold
Block header format error
Selection address error
Streaming tributary
Local buffer too small

3.8.4 Line Events

The Event Logger records the following events for all lines:

Automatic counters
Counters zeroed
Passive loopback

The Event Logger records the following Data Link LAPB line events:

Locally initiated state change
Remotely initiated state change
Block header format error

The Event Logger records the following Data Link Ethernet line events:

Initialization failed
Send failed
Collision detect check failed
Receive failed

The Event Logger records the following Physical Link line events:

Data set ready transition
Ring indicator transition
Unexpected carrier transition
Memory access error
Communications interface error
Performance error

3.8.5 Module Events

The Event Logger records the following Data Link X.25 protocol module events:

Restart
State change
Retransmit maximum exceeded
Block header format error
DTE up
DTE down

3.9 Event Parameters

This section describes the event parameters related to events described in Section 3.8. The user cannot directly control or observe these parameters. The Event Logger records parameters upon the occurrence of related events, if event logging of those parameters is enabled.

There are also events that relate to counters. These counters are those already described for nodes, circuits, lines, and modules. The event parameters are described in alphabetical order by layer, starting with the highest layer that maintains event parameters.

3.9.1 Network Management Layer

OPERATION

This parameter represents the operation performed, with the following values:

INITIATED
TERMINATED

REASON

This parameter indicates the reason the function aborted, with the following values:

Receive timeout
Receive error
Line state change by higher level
Unrecognized request
Line open error

SERVICE

This parameter represents the service type, with the following values:

LOAD
DUMP

STATUS

This parameter is the operation status, consisting of:

RETURN CODE   ERROR DETAIL   ERROR MESSAGE

where:

RETURN CODE

A standard NICE return code (Appendix F), with added interpretation, as follows:

REQUESTED
SUCCESSFUL
FAILED

ERROR DETAIL

A standard NICE error detail (Appendix F).

ERROR MESSAGE

A standard NICE optional error message (Appendix F).

3.9.2 Session Control Layer

ACCOUNT

This value contains any account information in a string of one to 39 characters. Account information is used for access control purposes.

DESTINATION PROCESS

This identifies the process to be connected to by Network Management. The identification consists of:

OBJECT TYPE - An object type number.

GROUP CODE - A group code number.

USER CODE - A user code number.

PROCESS NAME - A process name.

The Session Control specification specifies these values.

NEW STATE

This represents the new node state, with the following values:

ON
OFF
SHUT
RESTRICTED

OLD STATE

This represents the old node state, with the same values as for NEW STATE.

PASSWORD

This is the access control password field value used in connecting to the DECnet destination of an incoming call. If no password is specified, then none will be used. Password is a string of 1 to 39 characters.

REASON

This represents the reason for the state change, as follows:

Operator command
Normal operation

SOURCE NODE

This identifies the source node, where the source process resides which sent the session control protocol message which caused the event. The identification consists of a node address followed by a node name. The format is the same as for the node entity (Section 3.1).

SOURCE PROCESS

This identifies the source process on behalf of which the source node sent the session control protocol message which caused the event. The identification is the same as for DESTINATION PROCESS.

USER

This is a string of 1 to 39 characters identifying the user.

3.9.3 End Communication Layer

CURRENT FLOW CONTROL

This is the current flow control value (refer to the End Communication specification).

MESSAGE

This is the message received (NSP information only). The message consists of:

MESSAGE FLAGS - NSP message flags.

DESTINATION ADDRESS - Destination link address.

SOURCE ADDRESS - Source link address.

DATA - Message-type-dependent data.

SOURCE NODE

This is the identity of the node sending the ECL message. The source node consists of:

SOURCE NODE ADDRESS - Node address of the source node.

SOURCE NODE NAME - Name of the source node (optional).

3.9.4 Routing Layer

ADJACENT NODE

This is the identification of the adjacent node on the circuit.

HIGHEST ADDRESS

This is the highest reachable node address.

NODE

This is the identification of the node, in the same format as SOURCE NODE in the list of Session Control event parameters.

PACKET BEGINNING

This is the beginning of the packet.

PACKET HEADER

This is the packet header. For non-Ethernet packets, it consists of:

MESSAGE FLAGS - Message definition flags.

DESTINATION NODE ADDRESS - The address of the destination node.

SOURCE NODE ADDRESS - The address of the source node.

VISIT COUNT - The number of nodes the packet has visited.

For Ethernet packets, it consists of:

MESSAGE FLAGS - Message definition flags.

DESTINATION AREA - The area number of the destination node.

DESTINATION SUBAREA - The sub-area number of the destination node.

DESTINATION ETHERNET ADDRESS - The Ethernet address of the destination node.

SOURCE AREA - The area number of the destination node.

SOURCE SUBAREA - The sub-area number of the destination node.

SOURCE ETHERNET ADDRESS - The Ethernet address of the destination node.

NEXT AREA ROUTER - The number of the next area router.

VISIT COUNT - The number of nodes the packet has visited.

SERVICE CLASS - The packet service class.

PROTOCOL TYPE - The protocol type of the packet contents.

REASON

This is the reason for failure. The values are listed following Table 27 in section 7.13.

RECEIVED VERSION

This is the received version number, with the same format as for Network Management version (Section 3.1.1).

STATUS

This is the node status, with the following values:

REACHABLE
UNREACHABLE

3.9.5 Data Link Layer

BLOCK LENGTH

This is the received block length from header, in bytes.

BUFFER LENGTH

This is the buffer length, in bytes.

CAUSE

This represents the cause for the X.25 protocol module event. For detailed explanation of the value see the CCITT X.25 Recommendation.

DIAGNOSTIC

This represents the diagnostic for the X.25 protocol module event. For detailed explanation of the value see the CCITT X.25 Recommendation.

DISTANCE

This is the distance, in bit times, to a short or open cable on an Ethernet line.

DTE

This identifies the DTE associated with the X.25 protocol module event.

ETHERNET HEADER

This is the header of the Ethernet block. It includes destination address, source address, and protocol type.

FAILURE REASON

This is the reason for an Ethernet transmit or receive failure.

HEADER

This is the block header

NEW STATE

This is the new DDCMP state, with the following values:

HALTED
ISTRT
ASTRT
RUNNING
MAINTENANCE

NEW STATE

This represents the X.25 protocol module new state associated with event 5.12, with the same values as for the DDCMP NEW STATE.

OLD STATE

This is the old DDCMP state, with the same values as for NEW

STATE.

OLD STATE

This is the X.25 protocol module old state associated with event 5.12, with the same values as for the DDCMP NEW STATE.

PARAMETER TYPE

This is the Network Management parameter type of the parameter involved in the event.

PREVIOUS TRIBUTARY

This is the previously selected tributary address.

REASON

This is the reason for a state change.

RECEIVED TRIBUTARY

This is the received tributary address.

SELECTED TRIBUTARY

This is the selected tributary address.

TRIBUTARY STATUS

This is the tributary status, with the following values:

Streaming
Continued send after timeout
Continued send after deselect
Ended streaming

3.9.6 Physical Link Layer

DEVICE REGISTER

This is a copy of the contents of a single device register. When more than one, they are output in standard order.

NEW STATE

This represents the new modem control state, as follows:

OFF
ON

4 NETWORK CONTROL PROGRAM (NCP)

This section is divided into three parts. Section 4.1 describes the NCP functions. Section 4.2 provides rules for the operation of NCP, including such topics as input and output formatting and status and error messages. Section 4.3 presents a complete list of all the NCP commands as well as specific formats for the output on SHOW and LIST commands.

4.1 Network Control Program Functions

There are two types of NCP commands:

1. Internal commands. These are directed to NCP itself and cannot be sent to remote nodes. These are the SET and DEFINE EXECUTOR NODE node-id, CLEAR and PURGE EXECUTOR NODE, and SHOW QUEUE commands; the TELL prefix; and the EXIT command.

2. Commands that use the Network Management interface. These use the Network Management Listener, via the Network Information and Control Exchange (NICE) protocol, when sent across logical links to remote nodes. NCP commands directed to the local node have the option of either using the Network Management Listener, via the Network Management Access Routines and the NICE protocol, or of passing requests directly to the Local Network Management Function from the Network Management Access Routines. The method chosen is implementation-specific; however, passing requests internally is recommended.

The keyword ALL can be used with many of the NCP commands. In general, it means that the command should be executed for all parameters in the appropriate data base associated with the specified entity.

The NCP command language enables an operator to perform the following network functions:

- o Changing parameters (Section 4.1.1)
- o Gathering information (Section 4.1.2)
- o Down-line loading (Section 4.1.3)
- o Up-line dumping (Section 4.1.4)
- o Triggering bootstrap (Section 4.1.5)
- o Testing link and network (Section 4.1.6)

- o Zeroing counters (Section 4.1.7)

4.1.1 Changing Parameters

NCP can set or change many of the parameters described in Section 3.

Some examples of changing parameters are:

- o Setting a line state to ON
- o Changing a node name associated with a node address
- o Setting the routing cost fcr a line
- o Setting a node to be notified of certain logged events

Parameters may be set either as dynamic values in volatile memory using the SET command or as permanent values in a mass-storage default data base using the DEFINE command. The volatile data base is lost when the node shuts down; the permanent data base remains from one system initialization to the next. Parameters can be either status, such as line state, or characteristics that are determined by SET, DEFINE, CLEAR, and PURGE commands. Characteristics are static in the sense that once set, either at system generation time or by an operator, they remain constant until cleared or reset. Status consists of dynamic information (such as line state) that changes automatically when functions are performed.

Permanent values take effect whenever the permanent data base is re-read. The timing of the values taking effect is implementation-dependent. Volatile values take effect immediately.

Section 5.10 describes the internal operation for changing parameters.

4.1.2 Gathering Information

NCP can display current values for the parameters and counters described in Section 3. Examples of gathering information are:

- o Displaying the state of a line
- o Reading and then zeroing line counters
- o Displaying characteristics of all reachable nodes
- o Showing the status of all commands in progress at a node

Counters are error and performance statistics such as messages sent and received, time last zeroed, and maximum number of logical links in use. Section 5.11 describes the read information operation.

4.1.3 Down-line Loading

Down-line loading is the process of transferring a memory image from a file to a target system's memory. This requires that the executor, the node executing the command, have direct access to the link to the target. The file may be located at another remote node, in which case the executor uses its system-specific remote file access procedures. The executor supports or has access to a data base of defaults for a load request. Section 5.6 describes the down-line load operation in the Network Management layer.

4.1.4 Up-line Dumping

Up-line dumping is the process of transferring the dump of a memory image from a target system to a destination file. Section 5.7 describes the up-line dump operation.

4.1.5 Triggering Bootstrap

An operator can use NCP to trigger the bootstrap loader of an unattended remote target node. Section 5.8 describes the trigger bootstrap operation.

4.1.6 Testing Link and Network

Testing link and network can be accomplished by message looping at the line, circuit, and node levels. Testing requires receiving a transmitted message over a particular path that is looped back to the local node by either hardware or software.

Node level testing uses logical links and normal data link usage. The data links involved are in the ON state, and the Session Control, End Communication, and Routing layers are used.

During circuit level testing, a DDCMP circuit being tested is in the SERVICE state; normal usage is precluded. An X.25 circuit cannot be loop tested. An Ethernet circuit to be tested must be in the ON state and be owned by the LOOPER module. For all circuit tests, Network Management accesses the Data Link layer directly, bypassing intermediate layers.

During line loop testing, the line being tested is in the SERVICE state. As with the circuit loop test, normal usage is precluded. Network Management accesses the Data Link layer directly. A LAPB line loop is at the physical connection level, but is limited to hardware loopback only. Section 5.9 further describes line, circuit, and node testing.

4.1.7 Zeroing Counters

Using NCP, an operator can set module, line, circuit, and node counters to zero.

4.2 Network Control Program Operation

This section describes general rules concerning the operation on NCP.

Multiple parameters on SET, DEFINE, CLEAR, and PURGE commands are implementation optional. If they are allowed, either all must be successfully acted on, or none.

4.2.1 Specifying the Executor

Since a command does not have to be executed at the node where it is typed, the operator must be able to designate on what node the command is to be processed. The operator has two options for controlling this:

1. Specifying a default executor for a set of commands

2. Naming the executor with the command

At NCP start-up time, the default executor is the node on which NCP is running or the node that was previously defined with the DEFINE EXECUTOR NODE command. The default executor is changed using the SET, DEFINE, CLEAR, or PURGE EXECUTOR NODE commands.

With any command, the operator can override the default executor by specifying which node is to execute the command. This is accomplished by entering "TELL node-identification" as a prefix to the command. The specified node identification applies only to the one command and does not affect the default executor or any subsequent commands.

4.2.2 Program Invocation, Termination, and Prompting

The way NCP is invoked or terminated is system-dependent. If a name is used for the program, it must be "NCP." The EXIT command terminates NCP.

The following rules apply to the initial NCP prompt:

For an NCP that accepts only a single outstanding command, the prompt is always the same:

```
NCP>
```

For an NCP that accepts several outstanding commands where it is obvious that NCP is prompting, the prompt is:

    #n>

For the multiple-outstanding-command case where it is not obvious that NCP is prompting, the prompt is:

    NCP#n>

In any case, n is the command's request number, which will identify the output for the command.

An implementation that cannot integrate the request number with the prompt, can display the request number when the command is accepted.

4.2.3 Privileged Commands

Network and system planners must determine which commands should be limited to privileged users. The exact determination of privilege is an implementation-dependent function. Privilege is generally determined in a system-specific way according to the privileges of the local user or the access control provided at logical link connection time.

4.2.4 Input Formats

Command input is in the form of arguments delimited by tabs or blanks. Either a single or multiple tab or blank may be used to delimit arguments.

Null command lines. Null command lines will result in a command prompt being re-issued.

Node identification and access control. Nodes are identified by address or name. The primary identification is the address (a Session Control requirement). The keyword EXECUTOR can be substituted for NODE executor-node-identification. If a node identification represents a node to be connected to, access control information may be necessary or desired. If so, the access control follows the node identification, the maximum length of each field being 39 bytes. Specific systems may limit the amount of access control information they will accept. The format is:

Command	Entity	Parameter
LOOP NODE SET EXECUTOR NODE TELL	node-id	[USER user-id] [PASSWORD password] [ACCOUNT account]

where:

LOOP NODE node-id	Is an NCP command used to initiate a node loopback test. The access control applies only to the command.
SET EXECUTOR NODE node-id	Is an NCP command used to set the node identification and access control for the default executor node. The access control prevails until changed by another SET EXECUTOR command or a TELL or LOOP NODE command.
TELL node-id	Is an NCP command prefix used to pass one command and access control information to a specific node. The access control applies only to that one command.

For example:

TELL BOSS USER [211,1] PASSWORD secret ACCOUNT xyz CLEAR KNOWN LINES

SET EXECUTOR NODE 97 ACCOUNT xyz

String input. String input (every argument that is not a node name, keyword or number) is defined by the executor node and the length limitations of the NICE protocol. For consistency from one implementation to another, the following rules apply to NCP's parsing algorithm for these types of arguments:

- o Implementations will provide both a transparent and a non-transparent technique for specifying these arguments.
- o The transparent technique will act on any string of characters enclosed in quotation marks ("XXXXX"). A quote within the string will be indicated by a double quotation mark ("XXX""XX").
- o The non-transparent technique will act on any string of characters that does not contain blanks or tabs. An exception to this occurs where it is possible to recognize syntactically that blanks or tabs are not intended as delimiters.

Keywords. Implementations must accept keywords in their entirety. However, the user may abbreviate keywords when typing them in. The minimum abbreviation is system-specific.

The command formats specified in this document are to be the formats used for NCP input. They may be modified only in the sense that unsupported commands or options may be left out. It is permissible to prefix a command with an identifier such as OPR NCP. However, this prefix should not affect the remainder of the command syntax or semantics. Optional system-specific guide words such as TO or FOR can be added to NCP commands if they do not interfere with defined key words.

The NCP command language does not use a question mark as a syntactic or semantic element. The question mark is left available for use according to operating system conventions.

An implementation may recognize locally defined names for lines or accept other non-standard line identifications as string inputs.

4.2.5 Output Characteristics

The output format specified in this document is to be considered the basic pattern for all NCP output. Implementations may differ as long as common information is readily identifiable. The following example shows three commands and their resultant output. User-furnished information is underlined to distinguish it from the program output.

```
#23>LOAD NODE MANILA

#24>LOAD NODE TOKYO

#25
REQUEST #24; LOAD FAILED, LINE COMMUNICATION ERROR
SHOW QUEUE
REQUEST #25; SHOW QUEUE

REQUEST
NUMBER   EXECUTOR     COMMAND  STATUS

  21     6 (HNGKNG)   SHOW     COMPLETE
  22     6 (HNGKNG)   SET      COMPLETE
  23     6 (HNGKNG)   LOAD     IN PROGRESS
  24     6 (HNGKNG)   LOAD     FAILED
  25     N/A          SHOW     IN PROGRESS
#26>
REQUEST #23, LOAD COMPLETE
```

Passwords are not displayed. Instead, an ellipsis (...) indicates that a password is set. Section 4.3.8 contains output for requested information (SHOW and LIST commands).

4.2.6 Status and Error Messages

Status and error messages inform the NCP user of the consequence of a command entry. NCP gives each command a request number, which it displays with status and error messages. NCP displays status or error messages when the status of the command changes as long as the user does not begin to type a new command. The general form of status and error messages is:

```
REQUEST  n; [entity,] command status [,error-message]
```

where:

n	Is the command's request number.
entity	Is a specific entity.
command	Is a command indicator.
status	Is the status of the operation, one of COMPLETE, FAILED, or NOT ACCEPTED. If it is COMPLETE, there is no error-message. If it is FAILED or NOT ACCEPTED, there is an error-message.
error-message	Is the reason for a failure.

Commands that act on plural entities (for example, SET KNOWN LINES) have a separate status message for each individual entity and one for the entire operation. In this case, each entity is identified with its own status message.

In an NCP that allows only one command at a time, COMPLETE messages are not displayed, and the request number is not included. An example of output for a command that has failed follows:

```
LOAD FAILED, LINE COMMUNICATION ERROR
```

When a loop test succeeds on an Ethernet circuit and the user did not specify a physical address, the command output includes the ethernet-address of the responding system from the loopback assistant multicast group in the form:

```
PHYSICAL ADDRESS = ethernet-address
```

When a loop test fails, the error message contains added explanatory information, in the form either

```
UNLOOPED COUNT = n
```

or

```
MAXIMUM LOOP DATA = n
```

Where the unlooped count is the number of messages not yet looped when the test failed and maximum loop data is the maximum length that can be requested for the loop test data.

NCP prints unrecognized return codes or error details as decimal numbers. For example:

```
Request #5; SHOW failed, Management return #-34
SET FAILED, parameter not applicable, detail #2300
```

Error messages are either those from the set of NCP error messages in Appendix G, the NICE error returns in Appendix F or implementation specific.

4.3 Network Control Program Commands

This section describes NCP commands.

The following symbols are used in NCP command syntax descriptions:

[]	Brackets indicate optional input. In most cases these are the entity parameters and entity parameter options for a command.
UPPER CASE	Upper case letters signify actual input, that is keywords that are part of NCP commands.
lower case	Lower case letters in a command string indicate a description of an input variable, not the actual input.
spaces	Spaces between variables (not keywords) in a command string delimit parameters.
hyphens	Multi-word variables are hyphenated.
{ }	Braces indicate that any of the enclosed parameters are applicable.

All NCP commands have the following common syntax:

    verb entity entity-option(s)

where:

verb	Specifies the operation to be performed, such as SHOW or LOAD.
entity	Specifies the entity (component) to which the operation applies, such as LINE or KNOWN NODES.
entity-option(s)	Qualifies the command by providing further specific information.

4.3.1 SET and DEFINE Commands

These commands modify volatile and permanent parameters. The SET command modifies the volatile data base; the DEFINE command changes the permanent data base. Section 5.10 describes the internal change parameter operation.

The general form of the commands is:

```
SET         entity      parameter
DEFINE
```

Entity is one of the following:

CIRCUIT circuit-id
EXECUTOR
KNOWN CIRCUITS
KNOWN LINES
KNOWN LOGGING
KNOWN MODULES
KNOWN NODES
LINE line-id
LOGGING sink-type
MODULE module-name
NODE node-id

Parameter is one (or more, if allowed by the implementation) of the parameter options defined for the specified entity.

4.3.1.1 SET and DEFINE EXECUTOR NODE destination-node

The SET and DEFINE EXECUTOR NODE commands, processed by NCP, change the executor node for subsequent commands. Access control information may be supplied as described in Section 4.2.4.

4.3.1.2 SET and DEFINE KNOWN Entity Commands

These commands set volatile and permanent parameters for each one of the specified entities known to the system. The format is:

Command	Entity	Parameter
SET DEFINE	KNOWN plural-entity	ALL parameter

Plural-entity is one of CIRCUITS, LINES, LOGGING, MODULES, or NODES.

The parameters are the same as for the SET and DEFINE entity commands following. However, DEFINE KNOWN plural-entity ALL has no meaning. SET KNOWN plural-entity ALL loads all permanent entity parameters into the volatile data base.

4.3.1.3 SET and DEFINE CIRCUIT Commands

These commands control the setting of parameters for the circuit entity. Some of the parameters only apply to certain circuit types. Refer to Section 3.3.1.

The format of these commands is:

Command	Entity	Parameter
SET DEFINE	CIRCUIT circuit-id KNOWN CIRCUITS	ACTIVE BASE base ACTIVE INCREMENT increment ALL BABBLE TIMER milliseconds BLOCKING blocking-control CHANNEL channel-number COST cost COUNTER TIMER seconds DEAD THRESHOLD count DTE dte-address DYING BASE base DYING INCREMENT increment DYING THRESHOLD count HELLO TIMER seconds INACTIVE BASE base INACTIVE INCREMENT increment INACTIVE THRESHOLD count LINE line-id MAXIMUM BUFFERS count MAXIMUM DATA byte-count MAXIMUM RECALLS retry-count MAXIMUM ROUTERS number MAXIMUM TRANSMITS count MAXIMUM WINDOW block-count NUMBER call-number ORIGINATING QUEUE LIMIT queue-size OWNER owner-id POLLING STATE polling-state RECALL TIMER seconds ROUTER PRIORITY number SERVICE service-control STATE circuit-state TRANSMIT TIMER milliseconds TRIBUTARY tributary-address TYPE circuit-type USAGE usage-type

4.3.1.4 SET and DEFINE LINE Commands

These commands control the setting of parameters for the line entity. Some of the parameters are only applicable to certain line protocols. Refer to Section 3.5.1 The format of these commands is:

Command	Entity	Parameter
SET DEFINE	LINE line-id KNOWN LINES	ALL CLOCK clock-mode CONTROLLER controller-mode COUNTER TIMER seconds DEAD TIMER milliseconds

		DELAY TIMER milliseconds
		DEVICE device-specification
		DUPLEX duplex-mode
		HOLDBACK TIMER milliseconds
		MAXIMUM BLOCK byte-count
		MAXIMUM RETRANSMITS block-count
		MAXIMUM WINDOW block-count
		PROTOCOL protocol-name
		RECEIVE BUFFERS number
		RETRANSMIT TIMER milliseconds
		SCHEDULING TIMER milliseconds
		SERVICE service-control
		SERVICE TIMER milliseconds
		STATE line-state
		STREAM TIMER milliseconds

4.3.1.5 SET and DEFINE LOGGING Commands

This set of commands is used to control event sinks (where events are logged) and event lists (that control which events get logged). The command format is:

Command	Entity	Parameter	Qualifier
SET DEFINE	LOGGING sink-type	ALL	
		EVENT event-list KNOWN EVENTS	[source-qualifier] [sink-node]
		NAME sink-name STATE sink-state	

Section 3.3 describes source qualifiers, sink nodes, and sink types.

4.3.1.6 SET and DEFINE MODULE Commands

These commands vary considerably depending on the module named. The following descriptions take this into account by describing the options independently for each defined module identification.

4.3.1.6.1 SET and DEFINE MODULE CONSOLE Commands

These commands control the parameters necessary to the maintenance console. The format of these commands is:

Command	Entity	Parameter
SET DEFINE	MODULE CONSOLE	ALL RESERVATION TIMER seconds

4.3.1.6.2 SET and DEFINE MODULE LOADER Commands

These commands control the parameters necessary to the maintenance loader and dumper. The format of these commands is:

Command	Entity	Parameter
SET DEFINE	MODULE LOADER	ALL ASSISTANCE control

4.3.1.6.3 SET and DEFINE MODULE LOOPER Commands

These commands control the parameters necessary to the maintenance looper. The format of these commands is:

Command	Entity	Parameter
SET DEFINE	MODULE LOOPER	ALL ASSISTANCE control

4.3.1.6.4 SET and DEFINE MODULE CONFIGURATOR Commands

These commands control the parameters necessary to the maintenance configurator. The format of these commands is:

Command	Entity	Parameter	Qualifier
SET DEFINE	MODULE CONFIGURATOR	ALL SURVEILLANCE control	[CIRCUIT circuit-id] [KNOWN CIRCUITS]

If only one circuit is known, it is the default. If more than one circuit is known, the qualifier must be included.

4.3.1.6.5 SET and DEFINE MODULE X25-ACCESS Commands

These commands control the parameters necessary to the X.25 Gateway Access Routines. The format of these commands is:

Command	Entity	Parameter	Qualifier
SET	MODULE	ACCOUNT account	[KNOWN NETWORKS]
DEFINE	X25-ACCESS	ALL	[NETWORK network-name]
		NODE node-id	
		PASSWORD password	
		USER user	

If only one network is known, it is the default. If more than one network is known, the parameter must be included.

4.3.1.6.6 SET and DEFINE MODULE X25-PROTOCOL Commands

These commands control the parameters for the X.25 protocol control module. The format of these commands is:

Command	Entity	Parameter	Qualifier
SET	MODULE	ALL	[dte-qualifier]
DEFINE	X25-PROTOCOL		[group-qualifier]
		GROUP group-name group-options	
		CALL TIMER seconds	
		CLEAR TIMER seconds	
		DEFAULT DATA byte-count	
		DEFAULT WINDOW block-count	
		MAXIMUM DATA byte-count	
		MAXIMUM CLEARS retry-count	
		MAXIMUM RESETS retry-count	
		MAXIMUM RESTARTS retry-count	
		MAXIMUM WINDOW block-count	
		NETWORK network-type	
		RESET TIMER seconds	
		RESTART TIMER seconds	
		CHANNELS list	[dte-qualifier]
		COUNTER TIMER seconds	
		LINE line-id	
		MAXIMUM CIRCUITS count	
		STATE dte-state	

Dte-qualifier indicates to which local DTE address the command applies. It is of the form:

KNOWN DTES
DTE dte-address

If only one local DTE address is known, it is the default. If more than one local DTE address is known, the parameter must be included.

Group-qualifier indicates to which closed user group the command applies. It is of the form:

KNOWN GROUPS
GROUP group-name

Group-options are:

DTE dte-address
NUMBER group-number
TYPE group-type

Both DTE and NUMBER must be included, TYPE is optional.

4.3.1.6.7 SET and DEFINE MODULE X25-SERVER Commands

These commands control the parameters necessary to the X.25 Gateway Server. The format of these commands is:

Command	Entity	Parameter	Qualifier
SET DEFINE	MODULE X25-SERVER	ALL	[destination-qual]
		COUNTER TIMER seconds MAXIMUM CIRCUITS count	
		ACCOUNT account CALL MASK hex-value CALL VALUE hex-value GROUP group-name NODE node-id NUMBER dte-address OBJECT object-id PASSWORD password PRIORITY priority SUBADDRESSES range USER user	[destination-qual]

Destination-qual indicates to which destination the command applies. It is of the form:

KNOWN DESTINATIONS
DESTINATION destination-name

If only one destination is known, it is the default. If more than one destination is known, the parameter must be included.

4.3.1.7 SET and DEFINE NODE Commands

These commands set volatile or permanent parameters for a node. Certain parameters can be set only for the executor node or for adjacent nodes. See Table 20, Section 7.9. The format for the command is:

Command	Entity	Parameter
SET	EXECUTER	ADDRESS node-address
DEFINE	KNOWN NODES	ALL
	NODE node-id	AREA MAXIMUM COST number
		AREA MAXIMUM HOPS number
		BROADCAST ROUTING TIMER seconds
		BUFFER SIZE bytes
		CIRCUIT circuit-id
		COUNTER TIMER seconds
		CPU cpu-type
		DELAY FACTOR number
		DELAY WEIGHT number
		DIAGNOSTIC FILE file-id
		DUMP ADDRESS number
		DUMP COUNT number
		DUMP FILE file-id
		HARDWARE ADDRESS ethernet-address
		HOST node-id
		IDENTIFICATION id-string
		INACTIVITY TIMER seconds
		INCOMING TIMER seconds
		LOAD FILE file-id
		MAXIMUM ADDRESS number
		MAXIMUM AREA number
		MAXIMUM BROADCAST NONROUTERS number
		MAXIMUM BROADCAST ROUTERS number
		MAXIMUM BUFFERS number
		MAXIMUM CIRCUITS number
		MAXIMUM COST number
		MAXIMUM HOPS number
		MAXIMUM LINKS number
		MAXIMUM VISITS number
		NAME node-name
		OUTGOING TIMER seconds
		RETRANSMIT FACTOR number
		ROUTING TIMER seconds
		SECONDARY DUMPER file-id
		SECONDARY LOADER file-id
		SEGMENT BUFFER SIZE bytes
		SERVICE CIRCUIT circuit-id
		SERVICE DEVICE device-type
		SERVICE NODE VERSION node-version
		SERVICE PASSWORD password
		SOFTWARE IDENTIFICATION software-id
		SOFTWARE TYPE program-type
		STATE node-state
		SUBADDRESSES range
		TERTIARY LOADER file-id
		TYPE node-type

4.3.2 CLEAR and PURGE Commands

These commands clear parameters from the volatile and permanent data bases. The CLEAR command affects the volatile data base; the PURGE command affects the permanent data base. Not all parameters can be cleared individually. A cleared or purged parameter or entity identification is the same as one that has not been set or defined. If the parameter has a default value, it reverts to that value. The general form of the command is:

Command	Entity	Parameter
CLEAR PURGE	entity	parameter

The entities are the same as for the SET and DEFINE commands (Section 4.3.1).

4.3.2.1 CLEAR and PURGE EXECUTOR NODE Commands

The CLEAR EXECUTOR NODE command resets the executor to the command node. Note that CLEAR EXECUTOR does not return the executor to that defined in the permanent data base. The PURGE EXECUTOR NODE command redefines the executor in the permanent data base to the command node. Access control is reset as well.

4.3.2.2 CLEAR and PURGE KNOWN Entity Commands

These commands clear and purge parameters for all of the specified entities known to the system. The format of the command is:

Command	Entity	Parameter
CLEAR PURGE	KNOWN plural-entity	parameter

Plural-entity is one of CIRCUITS, LINES, LOGGING, MODULES, or NODES.

Parameter is one or possibly more of the parameters associated with the CLEAR and PURGE entity commands (following).

4.3.2.3 CLEAR and PURGE CIRCUIT Commands

The format of these commands is:

Command	Entity	Parameter
CLEAR PURGE	CIRCUIT circuit-id KNOWN CIRCUITS	ACTIVE BASE ACTIVE INCREMENT ALL BABBLE TIMER COUNTER TIMER DEAD THRESHOLD DYING BASE DYING INCREMENT DYING THRESHOLD INACTIVE BASE INACTIVE INCREMENT MAXIMUM BUFFERS MAXIMUM RECALLS MAXIMUM TRANSMITS OWNER RECALL TIMER TRANSMIT TIMER

4.3.2.4 CLEAR and PURGE LINE Commands

The format of these commands is:

Command	Entity	Parameter
CLEAR PURGE	LINE line-id KNOWN LINES	ALL COUNTER TIMER DEAD TIMER DELAY TIMER HOLDBACK TIMER MAXIMUM RETRANSMITS SCHEDULING TIMER STREAM TIMER

4.3.2.5 CLEAR and PURGE MODULE Commands

These commands vary considerably depending on the module named. The following descriptions take this into account by describing the options independently for each defined module identification.

4.3.2.5.1 CLEAR and PURGE MODULE X25-ACCESS Commands

The format of these commands is:

Command	Entity	Parameter	Qualifier
CLEAR PURGE	MODULE X25-ACCESS	ALL ACCOUNT PASSWORD USER	[KNOWN NETWORKS] [NETWORK network-name]

If only one network is known, it is the default. If more than one network is known, the network qualifier must be included.

4.3.2.5.2 CLEAR and PURGE MODULE X25-PROTOCOL Commands

The format of these commands is:

Command	Entity	Parameter	Qualifier
CLEAR PURGE	MODULE X25-PROTOCOL	ALL	[dte-qualifier] [group-qualifier]
		CALL TIMER CLEAR TIMER GROUP group-name group-options MAXIMUM CLEARS MAXIMUM RESETS MAXIMUM RESTARTS RESET TIMER RESTART TIMER	
		COUNTER TIMER LINE line-id	[dte-qualifier]

Dte-qualifier indicates to which local DTE address the command applies. It is of the form:

KNOWN DTES
DTE dte-address

If only one local DTE address is known, it is the default. If more than one local DTE address is known, the parameter must be included.

Group-qualifier indicates to which closed user group the command applies. It is of the form:

GROUP group-name
KNOWN GROUPS

Group-options is one or more of:

DTE dte-address
TYPE

4.3.2.5.3 CLEAR and PURGE MODULE X25-SERVER Commands

The format of these commands is:

Command	Entity	Parameter	Qualifier
CLEAR PURGE	MODULE X25-SERVER	COUNTER TIMER	
		ACCOUNT ALL CALL MASK CALL VALUE GROUP NUMBER PASSWORD PRIORITY USER	[destination-qual]

Destination-qual indicates to which destination the command applies. It is of the form:

KNOWN DESTINATIONS
DESTINATION destination-name

If only one destination is known, it is the default. If more than one destination is known, the parameter must be included.

4.3.2.6 CLEAR and PURGE LOGGING Commands

These commands, in conjunction with the SET and DEFINE LOGGING commands, control event sinks and event lists. The same general definitions (sink-node, sink-type, and source-qualifier) that apply to the SET LOGGING command apply here.

Command	Entity	Parameter	Qualifier
CLEAR PURGE	LOGGING sink-type	ALL NAME	
		EVENT event-list KNOWN EVENTS	[sink-node] [source-qualifier]

4.3.2.7 CLEAR and PURGE NODE Commands

These commands clear volatile (using CLEAR) or permanent (using PURGE) parameters for the node. Node identification can be either a node name or a node address, except for the CIRCUIT option where it must be a name. EXECUTOR may substitute for NODE executor-node-identification.

Command	Entity	Parameter
CLEAR PURGE	NODE node-id	ALL CIRCUIT COUNTER TIMER CPU DIAGNOSTIC FILE DUMP ADDRESS DUMP COUNT DUMP FILE HARDWARE ADDRESS HOST IDENTIFICATION INCOMING TIMER LOAD FILE NAME OUTGOING TIMER SECONDARY DUMPER SECONDARY LOADER SERVICE DEVICE SERVICE CIRCUIT SERVICE PASSWORD SOFTWARE IDENTIFICATION SOFTWARE TYPE TERTIARY LOADER

4.3.3 TRIGGER Commands

These commands trigger the bootstrap of the target node so that the node will load itself. It initiates the load of an unattended system. This command will work only if the target node either recognizes the trigger operation with software or has the necessary hardware in the correct state. Parameters specified with a command override the default parameters of the same type. If the circuit is an Ethernet circuit, the PHYSICAL ADDRESS must be included in the TRIGGER VIA command. The format of the commands is:

Command	Entity	Parameter
TRIGGER	NODE node-id	[PHYSICAL ADDRESS ethernet-address] [[SERVICE] PASSWORD password] [VIA circuit-id]
	VIA circuit-id	[PHYSICAL ADDRESS ethernet-address] [[SERVICE] PASSWORD password]

4.3.4 LOAD Commands

These commands initiate a down-line load. There are two variations. Node identification is either the node name or the node address of the target node. This command works only if the conditions for trigger are met, or if the target node has been triggered locally.

4.3.4.1 LOAD NODE Commands

These commands load the node identified on the circuit identified or on the circuit obtained from the volatile data base. Any parameter not specified in the command line defaults to whatever is specified in the volatile data base at the executor node.

Command	Entity	Parameter
LOAD	NODE node-id	[ADDRESS node-address]
		[CPU cpu-type]
		[FROM load-file]
		[HOST node-id]
		[NAME node-name]
		[PHYSICAL ADDRESS ethernet-address]
		[SECONDARY [LOADER] file-id]
		[SERVICE DEVICE device-type]
		[SERVICE NODE VERSION node-version]
		[[SERVICE] PASSWORD password]
		[SOFTWARE IDENTIFICATION software-id]
		[SOFTWARE TYPE program-type]
		[TERTIARY [LOADER] file-id]
		[VIA circuit-id]

4.3.4.2 LOAD VIA Commands

With these commands, the executor loads the target over the specified circuit, obtaining the node identification from the volatile data base if necessary. If the circuit is an Ethernet circuit, the PHYSICAL ADDRESS must be included in the command. The command format is:

Command	Entity	Parameter
LOAD	VIA circuit-id	[ADDRESS node-address]
		[CPU cpu-type]
		[FROM load-file]
		[HOST node-id]
		[NAME node-name]
		[PHYSICAL ADDRESS ethernet-address]
		[SECONDARY [LOADER] file-id]
		[SERVICE DEVICE device-type]
		[SERVICE NODE VERSION node-version]

		[[SERVICE] PASSWORD password]
		[SOFTWARE IDENTIFICATION file-id]
		[SOFTWARE TYPE program-type]
		[TERTIARY [LOADER] file-id]

4.3.5 DUMP Commands

These commands perform an up-line dump. Parameters not supplied default to those in the volatile data base at the executor node. There are two variations.

4.3.5.1 DUMP NODE Commands

The format for these commands is:

Command	Entity	Parameter
DUMP	NODE node-id	[[DUMP] ADDRESS number]
		[[DUMP] COUNT number]
		[PHYSICAL ADDRESS ethernet-address]
		[TO dump-file]
		[SECONDARY [DUMPER] file-id]
		[SERVICE DEVICE device-type]
		[[SERVICE] PASSWORD password]
		[VIA circuit-id]

4.3.5.2 DUMP VIA Commands

If the circuit is an Ethernet circuit, the PHYSICAL ADDRESS must be included in the command. The format for these commands is:

Command	Entity	Parameter
DUMP	VIA circuit-id	[[DUMP] ADDRESS number]
		[[DUMP] COUNT number]
		[PHYSICAL ADDRESS ethernet-address]
		[TO dump-file]
		[SECONDARY [DUMPER] file-id]
		[SERVICE DEVICE device-type]
		[[SERVICE] PASSWORD password]

4.3.6 LOOP Commands

These commands cause test blocks to loop back from the specified line or node. There are three variations, as described in the next three sections.

4.3.6.1 LOOP CIRCUIT Commands

These perform loop testing on a specific circuit. The optional parameters can be entered in any order. Parameters not specified default to their values in the permanent data base at the executor node. The command format is as follows:

Command	Entity	Parameter
LOOP	CIRCUIT circuit-id	[ASSISTANT NODE node-id] [ASSISTANT PHYSICAL ADDRESS ethernet-address] [COUNT count] [HELP help-type] [LENGTH length] [NODE node-id] [PHYSICAL ADDRESS ethernet-address] [WITH block-type]

If the circuit is an Ethernet circuit, and PHYSICAL ADDRESS is not included in the command, the Ethernet address used will be the loopback assistant multicast address. This will result in an output of the physical address that responded first.

HELP and ASSISTANT PHYSICAL ADDRESS can only be used with Ethernet circuits. If HELP is specified, an ASSISTANT PHYSICAL ADDRESS or ASSISTANT NODE must be included.

Ethernet circuits must be owned by MODULE LOOPER.

4.3.6.2 LOOP LINE Commands

The line loop performs loop testing on a specific line, which is unavailable for normal traffic during the test. The optional parameters can be entered in any order. Parameters not specified default to their values in the permanent data base at the executor node. The command format is as follows:

Command	Entity	Parameter
LOOP	LINE line-id	[COUNT count] [LENGTH length] [WITH block-type]

4.3.6.3 LOOP NODE Commands

A node loop will not interfere with normal traffic, but will add to the network load. The node loop can take place within one node or between two nodes. In the latter case, the remote node is the one specified (Figures 9 and 10, Section 5.9.1). EXECUTOR may be substituted for NODE executor-node-identification. The command format is as follows:

Command	Entity	Parameter
LOOP	NODE node-id	[access control]
		[COUNT count]
		[WITH block-type]
		[LENGTH length]

4.3.7 SHOW QUEUE Command

This command displays the status of the last few commands entered at the default executor. The number of commands displayed varies with each implementation. The executor for commands not sent across the network is shown as N/A (not applicable). Completed commands need not be displayed. Every command in progress must be shown in request number order. Implementations that do not allow multiple outstanding commands do not need this command.

An example of output follows:

```
REQUEST #13;  SHOW QUEUE

REQUEST
NUMBER              EXECUTOR   COMMAND   STATUS

9                   6 (HNGKNG)           LOAD       FAILED
10                  6 (HNGKNG)           SHOW       COMPLETE
11                  10(MANILA)           LOAD       IN PROGRESS
12                  6 (HNGKNG)           SET        COMPLETE
13                  N/A                  SHOW       IN PROGRESS
```

4.3.8 SHOW and LIST Commands

These commands are used to display information. The SHOW command displays information from the volatile data base. The LIST command displays information from the permanent data base. The format of the SHOW and LIST commands is:

Verb	Entity	Parameter	Qualifier
SHOW LIST	ACTIVE AREAS AREA area-number KNOWN AREAS	CHARACTERISTICS COUNTERS STATUS SUMMARY	
	ACTIVE CIRCUITS CIRCUIT circuit-id KNOWN CIRCUITS SIGNIFICANT CIRCUITS		[ADJACENT NODE node-id]
	ACTIVE LINES KNOWN LINES LINE line-id SIGNIFICANT LINES ACTIVE MODULES KNOWN MODULES		
	MODULE X25-ACCESS		[KNOWN NETWORKS] [NETWORK network-name]
	MODULE X25-PROTOCOL		[DTE dte-address] [GROUP group-name] [KNOWN DTES] [KNOWN GROUPS]
	MODULE X25-SERVER		[DESTINATION destination-name] [KNOWN DESTINATIONS]
	SIGNIFICANT MODULES ACTIVE NODES		
	ADJACENT NODES		[CIRCUIT circuit-id] [KNOWN CIRCUITS]
	EXECUTOR KNOWN NODES LOOP NODES NODE node-name SIGNIFICANT NODES		
	ACTIVE LOGGING KNOWN LOGGING LOGGING sink-type	CHARACTERISTICS EVENTS STATUS SUMMARY	[SINK NODE node-id] [KNOWN SINKS]
	MODULE CONSOLE	CHARACTERISTICS SUMMARY	

	MODULE CONFIGURATOR	STATUS SUMMARY	[KNOWN CIRCUITS] [CIRCUIT circuit-id]
	MODULE LOADER MODULE LOOPER		

KNOWN plural entities are all those known to the system, regardless of state. ACTIVE and SIGNIFICANT plural entities are subsets of KNOWN as defined in the glossary. When displaying plural nodes, the executor display is returned first, if it is included. Any loop nodes are returned last.

Sections 2 and 4.1.1 describe the information types, except SUMMARY. SUMMARY returns the most important information relating to the specified entity.

The tables in Section 7 specify the parameters and/or counters to be returned for each information type and entity.

Qualifiers can be placed either before or after the information type. When a qualifier is not specified in a command, the default is the "KNOWN" qualifier. An additional qualifier can be used for all entities:

TO alternate-output

This qualifier directs the output to an alternate output file or device (for example, a disk file or a line printer) rather than the default terminal display. The output is text in the same format it would have on the terminal. The format of the alternate output specification is system-dependent.

When there is no information to display in response to a SHOW command, the phrase "no information" is displayed in place of the data.

4.3.8.1 Information Type Display Format

All of the SHOW and LIST command information-type options have the same general output format. The header of that format is:

```
REQUEST  n; entity information-type AS OF dd-mon-yy hh-mm
```

For example:

```
REQUEST  21; KNOWN LINES STATUS AS OF 8-JUL-79 10:55

REQUEST  43; EXECUTOR NODE CHARACTERISTICS AS OF 10-SEP-79 10:56

REQUEST  45; KNOWN NODES SUMMARY AS OF 10-SEP-79 10:57
```

The requested information follows the header. The general format of the information is:

```
entity-type = entity-id

data
```

If the entity type is NODE, then one of EXECUTOR, REMOTE, or LOOP must precede it.

This information format repeats for each individual entity. A SHOW or LIST command with no information type should default to SUMMARY.

4.3.8.2 Counter Display Format

Counters are identified by standard type numbers as defined in Tables 6-8, 11-13, 18, and 19, Section 7. Counters are displayed in ascending order by type. The display format for counters is:

```
value description[, INCLUDING:]
         qualifier-1
              .
              .
              .
         qualifier-n
```

The value is the value of the counter, up to 10 digits for a 32-bit counter. It is a decimal number with no leading zeros. Zero values distinguish the case of no-counts from the case where a counter is not kept. If the counter has overflowed, it is displayed as the overflow value minus one, preceded by a greater-than sign. For example, an overflowed 8-bit counter would be displayed as ">254."

The description is the standard text that goes with the counter type as defined in Tables 6-8, 11-13, 18, and 19. If the counter type is not recognized, the description "COUNTER #n" is used, where n is the counter type number.

If the counter has an associated bit map, the word "including" is appended to the description, with a list of qualifiers. A qualifier is the standard text for the bit position in the bit map. A qualifier is displayed only if the corresponding bit is set. If the standard text for the bit is not known, the qualifier "QUALIFIER #n" is used, where n is the bit number.

For example:

```
REQUEST #21;  CIRCUIT COUNTERS AS OF 20-MAY-83 15:29

CIRCUIT = DUP-6

       532    TERMINATING PACKETS RECEIVED
       416    ORIGINATING PACKETS SENT
         0    TERMINATING CONGESTION LOSS
```

```
         400     TRANSIT PACKETS RECEIVED
         353     TRANSIT PACKETS SENT
          45     TRANSIT CONGESTION LOSS
       52379     BYTES RECEIVED
       41640     BYTES SENT
         963     DATA BLOCKS RECEIVED
         423     DATA BLOCKS SENT
           5     DATA ERRORS INBOUND, INCLUDING:
                   NAK'S SENT REP RESPONSE
           0     DATA ERRORS OUTBOUND
```

4.3.8.3 Tabular and Sentence Formats

Non-counter information permits two general formats. The first is easier to scan, the second is more extensible. The first is a tabular form, with each individual entity fitting on one line under a global header. Using this form, unrecognized parameter types are more clumsily handled and the amount of information per individual entity is limited to what will fit on one output line. The second is a sentence form. It adapts easily to a large number of parameters per individual entity and readily handles unrecognized parameter types.

In either form, the order of parameter output is the same in all implementations, even though in a particular implementation, some parameters may be unrecognized. The output format for unrecognized parameters is:

PARAMETER #n = value

where n is the decimal parameter number and value is the parameter value, formatted according to its data type.

Section 7 describes parameter types and their output order. In the sentence form of output, parameters that are logically grouped together should appear on the same line. Section 7 details these logical groupings.

The general output format of the data for tabular form is:

```
entity-type    parameter-type     parameter-type...

entity-id      parameter-value    parameter-value...
     .                .                  .
     .                .                  .
     .                .                  .
```

An example of output of the data in tabular form follows:

```
REQUEST  39; KNOWN CIRCUITS STATUS AS OF 18-SEP-78 15:20

CIRCUIT        STATE

BOSTON-0       ON
```

```
CHICAGO         OFF
CORUNNA         ON-LOADING
```

If NCP did not recognize an adjacent node parameter, the output would specify the type number of the parameter and the value according to the parameter data type. (See Tables 6 to 10, Section 7, for type numbers.)

The general output format of the data for sentence form is:

```
entity-type = entity-id

   par-type = par-value, par-type = par-value, ...
   par-type = par-value, ...
           .
           .
```

An example of output of the data for sentence form follows.

```
REQUEST #39;  KNOWN CIRCUITS STATUS AS OF 18-SEP-78  15:20

CIRCUIT = BOSTON-0

   STATE = ON

CIRCUIT = CHICAGO

   STATE = OFF

CIRCUIT = CORUNNA

   STATE = ON
```

The output format for the logging entity differs in the event display. For example, for the following command:

```
SHOW LOGGING CONSOLE SUMMARY KNOWN SINKS
```

A correct output would be

```
Logging Summary as of 7-MAR-79 10:55

Logging CONSOLE

     State = ON, NAME = CO0:

     Sink node = 15 (HALDIR), EVENTS =

          0.0,6
          Line KDZ-0-1.6, 3.6-7
          3.6-13

     Sink node = 16 (EOWYN), Events =

          0.0
```

Line KDZ-0-1.6, 6.0-1

4.3.8.4 Restrictions and Rules on Returns

The following restrictions and rules apply to returns on SHOW and LIST entity information type commands.

1. Node parameters. The parameters displayed for the SHOW and LIST NODE commands depend on which node is specified. Table 20, Section 7, indicates these restrictions. The keywords EXECUTOR, REMOTE or LOOP must precede NODE in a display of a node to clarify what is displayed.

2. Line and circuit states. The returns on the SHOW and LIST LINE/CIRCUIT STATUS commands must show the link substate as well as the state.

3. Loop nodes. Information for a single loop node is returned when requested by the loop node name. Information for multiple loop nodes is returned at the end of the display for KNOWN or ACTIVE NODES. It is the exclusive display for LOOP NODES.

4. Counters. COUNTERS can only be displayed with the SHOW commands, and with line, circuit, module, or node entities.

5. Events. EVENTS applies only to the logging entity. Sink node identification must be address and name (if a name exists), even for the executor.

4.3.9 ZERO Commands

These commands cause a specified set of counters to be set to zero. A zero command generates a counters zeroed event that causes counters to be logged before they are zeroed. The counters zeroed are those the executor node supports for the specified entity. The command format is:

Command	Entity	Parameter
ZERO	CIRCUIT circuit-id	[COUNTERS]
	EXECUTOR	
	LINE line-id	
	KNOWN CIRCUITS	
	KNOWN LINES	
	KNOWN MODULES	
	KNOWN NODES	
	MODULE module-name	
	NODE node-id	

For module X25-PROTOCOL the following qualifiers are added:

KNOWN DTES
DTE dte-address

If only one local DTE address is known, it is the default. If more than one local DTE address is known, the parameter must be included.

4.3.10 EXIT Command

This command terminates an NCP session.

5 NETWORK MANAGEMENT OPERATION

This section specifies the functionally-correct operation of Network Management. Implementations may use algorithms other than those contained herein, as long as the function is as specified here. The operations described in this section are:

NICE Access Routines and Listener (Section 5.1)
Local Network Management Functions (Section 5.2)
Link Watcher (Section 5.3)
Data Link Service Functions (Section 5.4)
Event Logger (Section 5.5)
Down-Line Load (Section 5.6)
Up-Line Dump (Section 5.7)
Trigger Bootstrap (Section 5.8)
Loop Test (Section 5.9)
Change Parameter (Section 5.10)
Read Information (Section 5.11)
Zero Counters (Section 5.12)
Loopback Mirror (Section 5.13)
NICE Logical Link Handling (Section 5.14)
Algorithm for Accepting Version Numbers (Section 5.15)
Return Code Handling (Section 5.16)

For Ethernet circuits, there is a special algorithm necessary in some cases. It is needed for Trigger, Dump, or Load Operations when the Ethernet address is not contained in the request. In this case, there are two possible Ethernet addresses: the hardware address from the node data base, or the expansion of the DNA node address.

To choose which of these addresses to use, the executor runs a normal, single message loop test to the hardware address. If this does not succeed within 2 seconds, the executor aborts it and tries the same loop test to the expanded DNA address. If this does not succeed within 2 seconds, the executor repeats the entire process 3 more times. If at the end of this procedure no response has been received, the original request fails with a communication error. If a response is received, that Ethernet address is used to satisfy the original request.

5.1 NICE Access Routines and Listener

The Network Management Access Routines receive NICE commands from the Network Control Program (NCP) and user programs. Network Management Access Routines pass NICE messages to the remote or local Network Management Listener via logical links. They also pass local function requests to the Local Network Management Functions. The Network Management Listener receives NICE command messages via logical links from the Network Management Access Routines in the local node or in other nodes.

The method used for processing Network Management functions within a single node is implementation-dependent. The Network Management

Access Routines can pass all local function requests to the Local Network Management Functions. Alternatively, the access routines can pass NICE messages to the Network Management Listener via a logical link. The latter method cannot be used for functions, such as turning the network on, that occur before a logical link is possible.

5.2 Local Network Management Functions

The Local Network Management Functions receive the following types of requests from other modules:

- System-independent function requests from the local NCP via the Network Management Access Routines.
- NICE function requests from other nodes via the Network Management Listener.
- NICE function requests from the local node via the Network Management Listener.
- Automatically-sensed service requests from the Link Watcher.

The Local Network Management Functions have the following interfaces to other modules or layers:

- Maintenance Functions - The Local Network Management functions have interfaces to the Maintenance Functions as described in the DNA Low Level Maintenance Operation specification.
- Link Service Functions - The Local Network Management Functions have a control interface to the Data Link Service Functions for setting and changing circuit and line states. The Local Network Management Functions have a "user" interface to the Data Link Service Functions for handling functions that are necessary for service functions (such as up-line dumping, down-line loading, and line level testing) to be performed.
- Control interfaces to lower layers - The Local Network Management Functions interface with lower layers directly for control and observation of lower level counters and parameters. An example of such an interface is examining a node counter.
- Function requests to lower layers and to local operating system - The Local Network Management Functions pass such function requests as file access, node level loopback, and timer setting to the application layer or to the local operating system in the form of system-dependent calls.

- o Event logging - The Local Network Management Functions interface with the Event Logging module in order to set event logging parameters that control such things as which events are logged and at what sink node they are logged.

Sections 5.6 to 5.16 supply algorithms for handling Network Management function requests.

5.3 Link Watcher

The Link Watcher module senses data link level service requests to up-line dump or load coming on an exclusive maintenance link from an adjacent node.

The Line Watcher senses a request by calling the Data Link Service Functions. Using parameters from that message, the Link Watcher then determines the request type and calls the Local Network Management Functions to accomplish the request.

The algorithm for implementing the Link Watcher is as follows:

```
Call Link Service Functions to get Data Link Service request for
     link
IF Link Service requested
    Set link state to ON-AUTOSERVICE (Local Network Management
         Functions)
    Determine function needed
    Call Network Management Functions to perform needed
         function(s)
    Reset link state to ON (Local Network Management Functions)
ENDIF
```

Section 5.10 describes the algorithms for setting and resetting link states for the Link Watcher.

5.4 Data Link Service Functions

The Data Link Service Functions provide exclusive maintenance link state changing and link handling services. They are used for functions requiring a direct interface to the Data Link layer. The functions that use the Data Link Service Functions are:

- o Down-line load (Section 5.6)
- o Up-line dump (Section 5.7)
- o Trigger bootstrap (Section 5.8)
- o Link test (Section 5.9.2)

1. Active at the executor node
2. Passive at the target node (for unattended system)

- o Set link state (Section 5.10)

The Data Link Service Functions provide the following services:

- o Condition a node to be dumped, loaded or have a loopback test performed. This state of the target node is called service slave mode, a mode in which the entire processor is taken over. Control rests with the executor.

- o Notify a higher level that active link services (load, dump) are needed.

- o Provide transmit/receive interface to higher level for active link services.

Section 3.6 describes line and circuit states and substates.

5.4.1 States and Substates

To arbitrate the use of the link, Data Link Service Functions maintain states and substates. Table 4, following, shows these as well as corresponding link states and substates displayed with the NCP SHOW CIRCUIT/LINE STATUS command. Table 4 also shows related Data Link Service functions.

The link can go from any substate to service slave mode.

Table 4
Line Service States, Substates and Functions and
Their Relationship to Link States

Link State	Link Substate	Link Service State	Link Service Substate	Link Service Function in Progress or Allowed
ON		passive	idle	Pass message to higher level
ON	-STARTING	passive	idle	Pass message to higher level
ON	-REFLECTING	passive	reflecting	Passive loopback
ON	-LOADING	open	loading	Receive and transmit loading messages
ON	-DUMPING	open	dumping	Receive and transmit dumping messages
ON	-TRIGGERING	open	triggering	Receive and

				transmit triggering messages
ON	-LOOPING	open	looping	Receive and transmit looping messages
ON	-AUTOSERVICE	closed	idle	Pass message to higher level
ON	-REFLECTING	closed	reflecting	Passive loopback
ON	-AUTOLOADING	open	loading	Receive and transmit loading messages
ON	-AUTODUMPING	open	dumping	Receive and transmit dumping messages
On	-AUTOTRIGGERING	open	triggering	Receive and transmit triggering messages
SERVICE		closed	idle	Pass message to higher level
SERVICE	-REFLECTING	closed	reflecting	Passive loopback
SERVICE	-LOADING	open	loading	Receive and transmit loading messages
SERVICE	-DUMPING	open	dumping	Receive and transmit dumping messages
SERVICE	-TRIGGERING	open	triggering	Receive and transmit triggering messages
SERVICE	-LOOPING	open	looping	Receive and transmit looping messages
OFF		off	idle	-

5.4.2 Priority Control

The Data Link Service Functions must make sure that higher priority functions take over, and that lower priority functions are resumed when higher priority functions are complete. The priorities are as follows from highest (1) to lowest (5):

1. Enter service slave mode (MOP primary mode) for passive line loopback, receiving down-line load, sending up-line dump, and transferring control. Control rests with the executor node. Some implementations may require hardware support.

2. No line operation (off state). In some implementations, this is the first priority.

3. Active service functions (send down-line load, trigger bootstrap, receive up-line dump, perform active line loopback).

4. Passive line loopback.

5. Normal operation (line available for use by owner).

5.4.3 Link State Algorithms

The algorithms that follow are a model for implementation of the Data Link Service states. If these algorithms are followed, the proper state transitions will take place. The algorithms refer to Data Link maintenance mode. This is a Data Link layer mode (DDCMP functional specification).

```
Set link state to off:

     Call Data Link to halt link
     Set substate to idle

Set link state to passive:

     IF link state is off or closed

          IF substate is not reflecting
               Set substate to idle
          ENDIF
     ELSE
          Fail
     ENDIF

Set link state to closed:

     IF link state is off, passive, or open
          IF link state is off or passive and substate is not
             reflecting
     Call Data Link to set link mode to maintenance
               Set substate to idle
          ENDIF
     ELSE
          Fail
     ENDIF

Set link state to open:

     IF link state is passive or closed
          Call Data Link to set link mode to maintenance
          IF substate is reflecting
               Terminate passive loopback
          ENDIF
          Record substate according to open parameter
```

```
ELSE
          Fail
     ENDIF
```

NOTE

The Data Link call to set the link mode to maintenance is a single operation that will succeed regardless of the state in which Data Link has the line when the call is issued.

5.4.4 Link Handling Functions

The link handling services of the Data Link Service Functions and the algorithms for implementing them follow.

1. Handling link in passive state (for entering service slave mode, passive loopback and passing message to a higher level):

```
WHILE link state is passive
     Call Data Link to see if link mode has gone to
     maintenance
     IF link mode has gone to maintenance
          Call Data Link to receive the service message
          IF enter service slave mode message
               Enter service slave mode
          ELSE IF loop data message
               Perform passive loopback algorithm
          ELSE IF looped data message
               Ignore
          ELSE
               On request, pass message to higher level
          ENDIF
          IF link state is still passive
               Call Data Link to halt link
          ENDIF
     ENDIF
ENDWHILE
```

2. Handling link in closed state (for entering service slave mode and performing passive loopback):

```
WHILE link state is closed
     Call Data Link to receive message
     IF enter service slave mode message
          Enter service slave mode
     ELSE IF loop data message
          Perform passive loopback algorithm
     ENDIF
ENDWHILE
```

3. Handling link in open state (for entering service slave mode, receiving a message, and transmitting a message):

```
WHILE link state is open
     IF transmit requested
          Call Data Link to transmit message
     ELSE IF receive requested
          IF data overrun recorded
               Return data overrun error
          ELSE
               Post receive requested
          ENDIF
     ENDIF
     Call Data Link to receive message
     IF enter service slave mode message
          Enter service slave mode
     ELSE
          IF receive posted
          Return message
          ELSE
               Record data overrun
          ENDIF
     ENDIF
ENDWHILE
```

4. Handling passive link loopback (passive at the remote or target node):

```
(Initial message already received)
Set substate to reflecting
WHILE substate is reflecting
     IF loop data message
     Call Data Link to transmit looped data message with
     received data
     Call Data Link to receive a message
     IF timeout or start received or error or loopback
          terminated
             Set substate to idle
     ENDIF
  ELSE
     Set substate to idle
  ENDIF
ENDWHILE
```

5.5 Event Logger

This module, diagrammed in Figure 6, following, records events that may help maintain the system, recover from failures, and plan for the future. Events originate in each of the DNA layers. Section 7.12 specifies event parameters. A system manager controls event recording with the SET LOGGING EVENT event-list command. The event list entered may require the Event Logger to filter out the recording of certain events.

```
                                     .---------------.
                                     | Event Monitor |
                                     `---------------'
                                               A_ _ _ _ _ _ _ _ _ .
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - |- - - - -
NETWORK MANAGEMENT LAYER                                          |
                                                                  |
           ----- .----------.   .-----------.   .-------.   .-----------.
from Event   :   | Event    |-->| Event     |-->| Event |-->| Event     |
Transmitters :   | Receiver |.->| Recorder  |--.| Queue |   | Monitor   |
           ----- `----------'|  `-----------'. |`-------'   | Interface |
             A               |               | |            `-----------'
     .-------'               |               | |
     |                       |               | |
     `--processed events----.|               | |  .--------.  .--------.
                            ||               | |  | Event  |  `--------'
                            ||               | `->| Queue  |->: Event  :
               .-----------.v|               |    `--------'  : File   :
               |           o-'               |                `--------'
               | Event     o------------.    |
               | Processor |    :       |    |    .--------.  .---------.
               |           o-------.    |    |    | Event  |  | Event   |
               `-----------'\      |    |    `--->| Queue  |->| Console |
                     A     Event   |    |         `--------'  `---------'
                     |     Filters |    |
                     |             |    |
                     |             |    |    .--------.  .-------------.
                     |             |    |    | Event  |  | Event       |
                     |             |    `--->| Queue  |->| Transmitter |---
     .-------.       |             |         `--------'  `-------------'
  .->| Event |------>|             |             :             :     to
 /   | Queue |       |             |             :             :   event
(    `-------'       |             |             :             :  receivers
 \                   |             |         .--------.  .-------------.
  raw events         |             |         | Event  |  | Event       |
                     |             `-------->| Queue  |->| Transmitter |---
                     |                       `--------'  `-------------'
- - - - - - - - - - -|- - - - - - - - - - - - - - - - - - - - - - - - - -
LOWER LAYERS         |        .--------------.
                     |<-------| Event Queues |<-------raw events
                              `--------------'
```

Figure 6. Event Logging Architectural Model

DECnet Event Logging is specified to meet the following goals:

- Allow events to be logged to multiple sink nodes including the source node.
- Allow an event to be logged to multiple logging sinks or any sink node.

- o Allow the definition of subsets of events for a sink or a node by event type and source node.
- o Include the following logging sinks: console, file, and monitor program.
- o Allow sharing of sinks between network event logging and local system event logging.
- o Minimize processing, memory, and network communication required to provide event logging.
- o Never block progress of network functions due to event logging performance limitations.
- o Minimize loss of event logging information due to resource limitations.
- o Record loss of event logging information due to resource limitations.
- o When required due to resource limitations, discard newer information (which can often be regained by checking current status) in favor of older.
- o Minimize impact of an overloaded sink on other sinks.
- o Standardize content and format of event logging information to the extent practical, providing a means of handling system specific information.
- o Allow independent control of sinks at sink node, including sink identification and sink state. Sink states include use of sink, non-use of sink, and temporary unavailability of sink.

5.5.1 Event Logger Components

As shown in Figure 6, the Event Logger consists of the following components, described in this section:

- o Event queue
- o Event processor
- o Event transmitter
- o Event receiver
- o Event recorder

- o Event console
- o Event file
- o Event monitor interface
- o Event monitor

Event queue -- There are several event queues (Figure 6). Each one buffers events to be recorded or transmitted, and controls the filling and emptying of the queue.

An event queue component has the following characteristics:

- o It buffers events on a first-in-first-out basis.
- o It fills a queue with one module; empties it with another.
- o It ensures that the filling module does not see an error when attempting to put an event on the queue.

Since event queues are not of infinite length, events must be lost. The filling module must record the loss of an event as an event, not as an error because of the third characteristic above. This event is called an "events-lost" event. An implementation requires the following algorithm at each event queue:

```
IF queue is full
    Discard the event
ELSE
    IF this event would fill the queue
        Discard the event
        IF last event on queue is not "events-lost"
            Queue an "events-lost" event (which fills the queue)
        ENDIF
    ELSE
        Queue the event
    ENDIF
ENDIF
```

The event queue component handles "events-lost" events according to the following rules.

1. Consider such events "raw" for raw event queues and "processed" for processed event queues.
2. Flag such events for the sink types of the lost events.
3. Time stamp such events with the time of first loss.
4. Filter such events only if all events for the queue are also filtered. Specifically, this means that event 0.0 cannot be filtered unless all other events are filtered.

Event Processor -- This component performs the following functions:

1. Scans the lower level event queues, collecting raw event records.

2. Modifies raw events into processed events. Raw events contain the following fields:

EVENT CODE	ENTITY IDENTIFICATION	DATA

   Processed events contain the following fields:

EVENT CODE	SOURCE NODE ID	SINK FLAGS	ENTITY NAME	DATE AND TIME STAMP	DATA

3. Compares the processed events with the event filters for each defined sink node, including the executor. Following are the characteristics of the filters used to control event logging:

   - o The event source node maintains all filters.

   - o Each event sink node has a separate set of filters at the source node.

   - o Each sink node set of filters contains a set of filters for each sink (monitor, file, or console).

   - o Each sink node set of filters contains a set of global filters, one global filter for each event class. It also contains one or more specific filters, each for a particular entity within an event class.

   - o Each filter contains one bit for each event type within the class. The bit reflects the event state: SET if the event is to be recorded, CLEAR if it is not.

   - o The filtering algorithm sees first if there is a specific filter that applies to the event by looking for an event mask whose source qualifier matches the entity name field. If so, the algorithm uses the specific filter. If not, the algorithm uses the global filter for the class.

   - o Commands from higher levels create and change filters using the EVENTS event-list option. When the specific filters match the global filter, the event processor deletes specific filters.

   - o Although the filters are modeled in the event processor, in some implementations, to reduce information loss or for efficiency reasons, it may be necessary to filter raw events before they are put into the first event queue. A reasonable, low-overhead way to implement this is by providing an event on/off switch at the low level. The high level can turn this switch off if the event is filtered out by all possible filters. This avoids a complex filter data base or search at the low level, but prevents flooding the low level event queue with unwanted events.

4. Passes events not filtered out to the event recorder for the executor or to the appropriate event queue for other sink nodes.

Event Transmitter. Using a logical link, this component transmits event records from its queue to the event receiver on its associated sink node.

Event Receiver. This component receives event records over logical links from event transmitters in remote event source nodes. It then passes them to the event recorder.

Event Recorder. This module distributes events to the queues for the various implementations, event sinks according to the sink flags in the event records.

Event Console. This is the event logging sink at which human-readable copies of events are recorded.

Event File. This is the event logging sink at which machine-readable copies of events are recorded. To Network Management, it is an append-only file.

Event Monitor Interface. This interface makes events available to the Network Management Functions for reading by higher levels.

Event Monitor. This user layer module is an "operator's helper." It monitors incoming events by using the Network Management Access Routines and may take action based on what it has seen. Its specific responsibilities and algorithms are undefined for the near term.

5.5.2 Suggested Formats for Logging Data

Following are suggested text formats for logging data. System specific variations that do not obscure the necessary data or change standard terminology are allowed.

The date field in the output is optional if it is obvious from the context of the logging output.

Milliseconds can be used in the event time data if it is possible to do so. If not supported, this field will not be printed. It is possible for two times given the same second to be logged and printed out of order.

General format:

```
EVENT TYPE class.type[, event-text]
```

```
FROM NODE address[(node-name)]OCCURRED [dd-mon-yy]hh:mm:ss:[.uuu]
[entity-type[entity-name]]
[data]
```

For example:

```
Event type 4.7, Packet ageing discard
From node 27 (DOODAH), occurred 9-FEB-79 13:55:38
Packet header = 2 23 91 20

Event type 0.3, Automatic line service
From node 19 (ELROND), occurred 9-FEB-79 16:09:10.009
Line KDZ-0-1.3, Service = Load, Status = Requested
```

Or, on a node that does not recognize the events:

```
Event type 4.7
From node 27, occurred 9-FEB-79 13:55:38
Parameter #2 = 2 23 91 20

Event type 0.3
From node 19, occurred 9-FEB-79 16:09:10.009
Line KDZ-0-1.3, Parameter #0 = 0, Parameter #1 = 0
```

5.6 Down-line Load Operation

The down-line capability allows the loading of a memory image from a file to a target node. The file may reside at the executor node or at another node. Any node can initiate the load.

The requirements for a down-line load are as follows:

- o The target node must be directly connected to the executor node via a physical link. The executor node provides the data link level access.

- o The target node must be running a minimal cooperating program (refer to the DNA Low Level Maintenance specification). This program may be a primary loader from a bootstrap ROM. The down-line load procedure may actually involve loading a series of programs, each of which calls the next program until the operating system itself is loaded. The initial program request information determines the load file contents.

- o The direct access link involved must be in the ON or SERVICE state.

- o The executor must have access to the file. The location of the file can be either specified in the load request or looked up by the Maintenance Functions.

  Maintenance Function modules are used to obtain local files. Remote files are obtained via remote file access techniques.

(Refer to the DAP functional specification.)

Figures 7A and 7B, following, show local and remote file access for a down-line load.

- o The executor must have access to a node data base, which can be either local or remote.
- o The target node must be able to recognize the trigger operation with software or hardware or must be triggered locally.

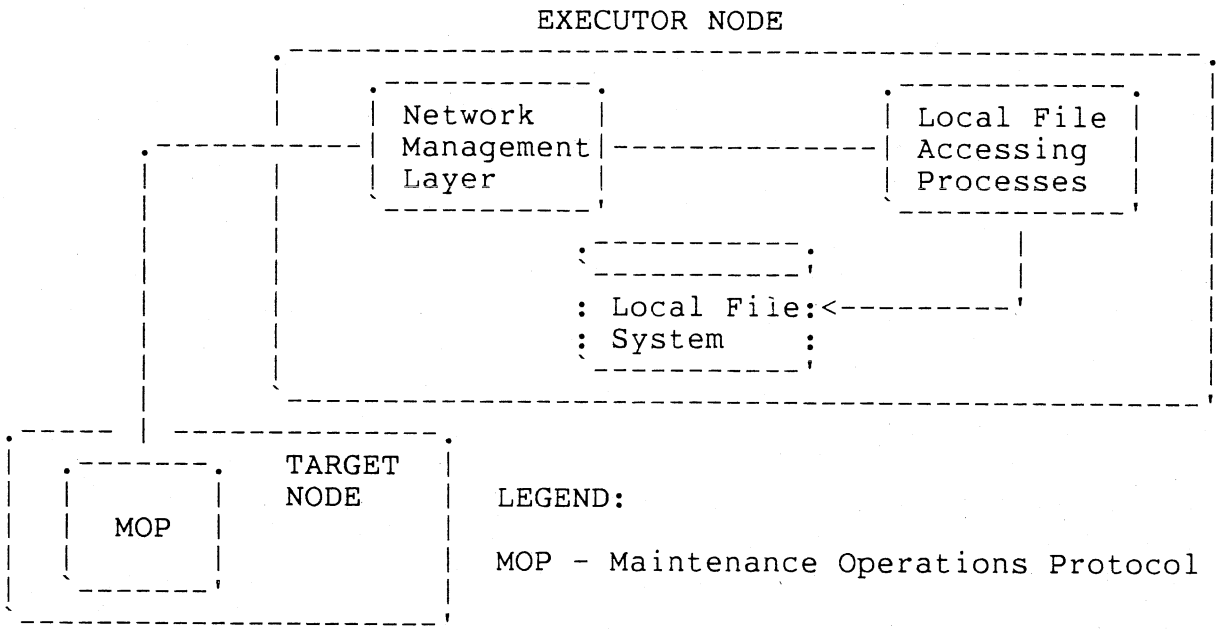
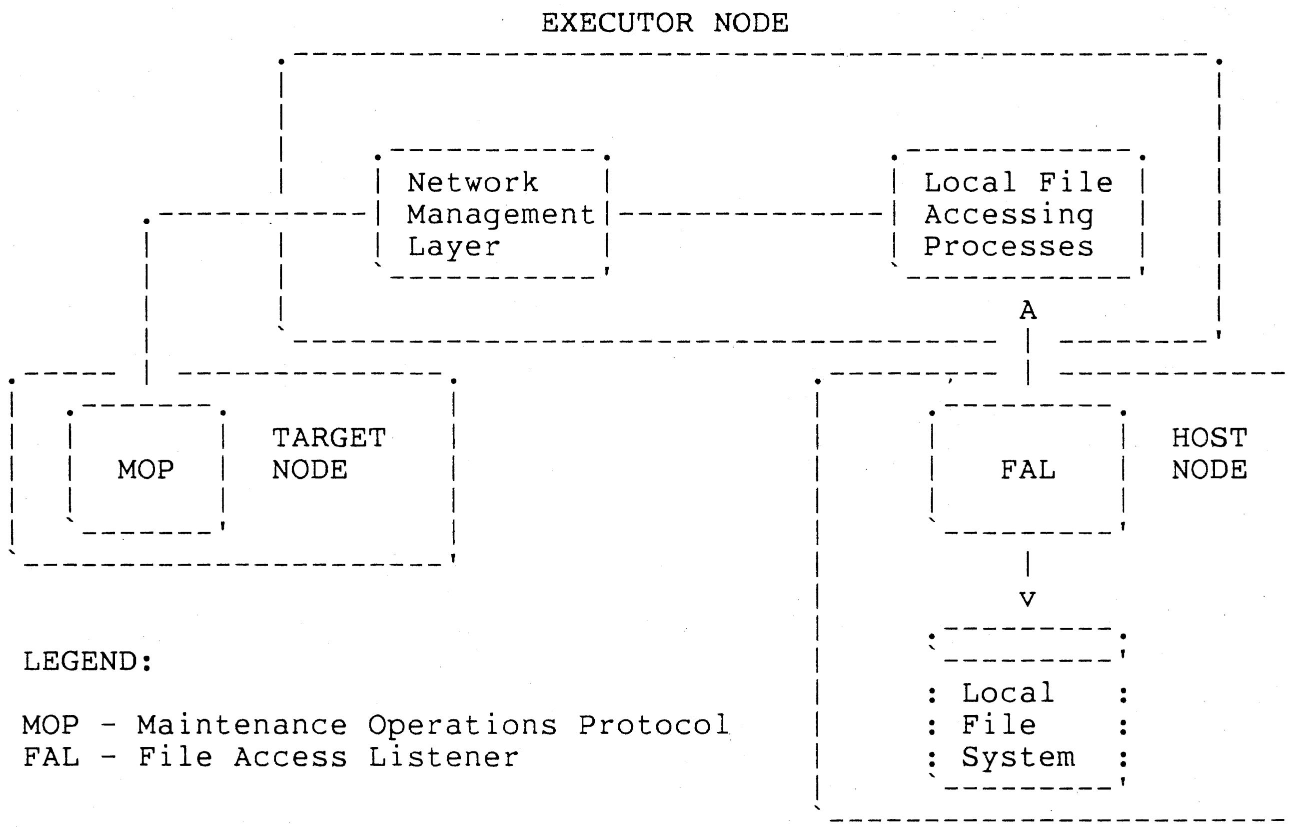


Figure 7A. Down-Line Load Local File Access Operation



LEGEND:

MOP - Maintenance Operations Protocol
FAL - File Access Listener

Figure 7B. Down-Line Load Remote File Access Operation

Either the target or executor node (or a remote command node) can initiate a down-line load. The target node initiates the load by triggering its boot ROM. The executor node initiates the load with either a trigger command or a load request. If the executor does not have the initial program request or the target does not respond to the attempt to load it, the executor should trigger the target.

Once the target is triggered, it requests the down-line load. The target node may be programmed to request the load over the line on which the trigger message came. Or, the target node could request the load from another executor. The Link Watcher at the executor senses the first program request from the target node (usually a request for the secondary loader, described below). Or, if the operation was initiated by a Network Management load request, the program request is received as a response to that request. Figures 8A and 8B, following, show the down-line load request operation.

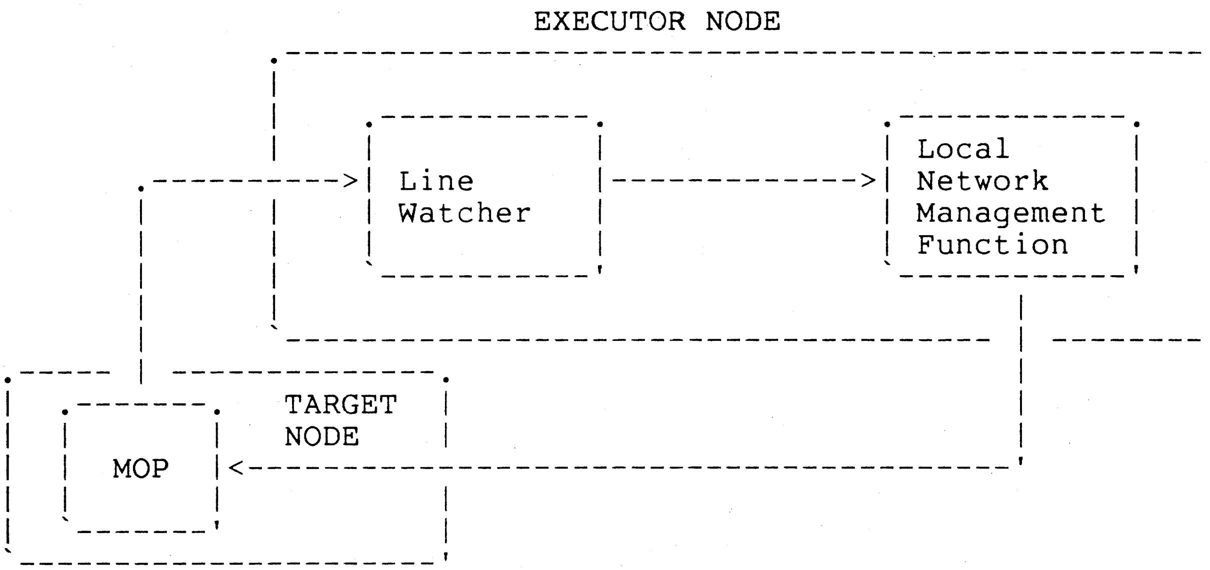


Figure 8A. Target-Initiated Down-Line Load Request Operation

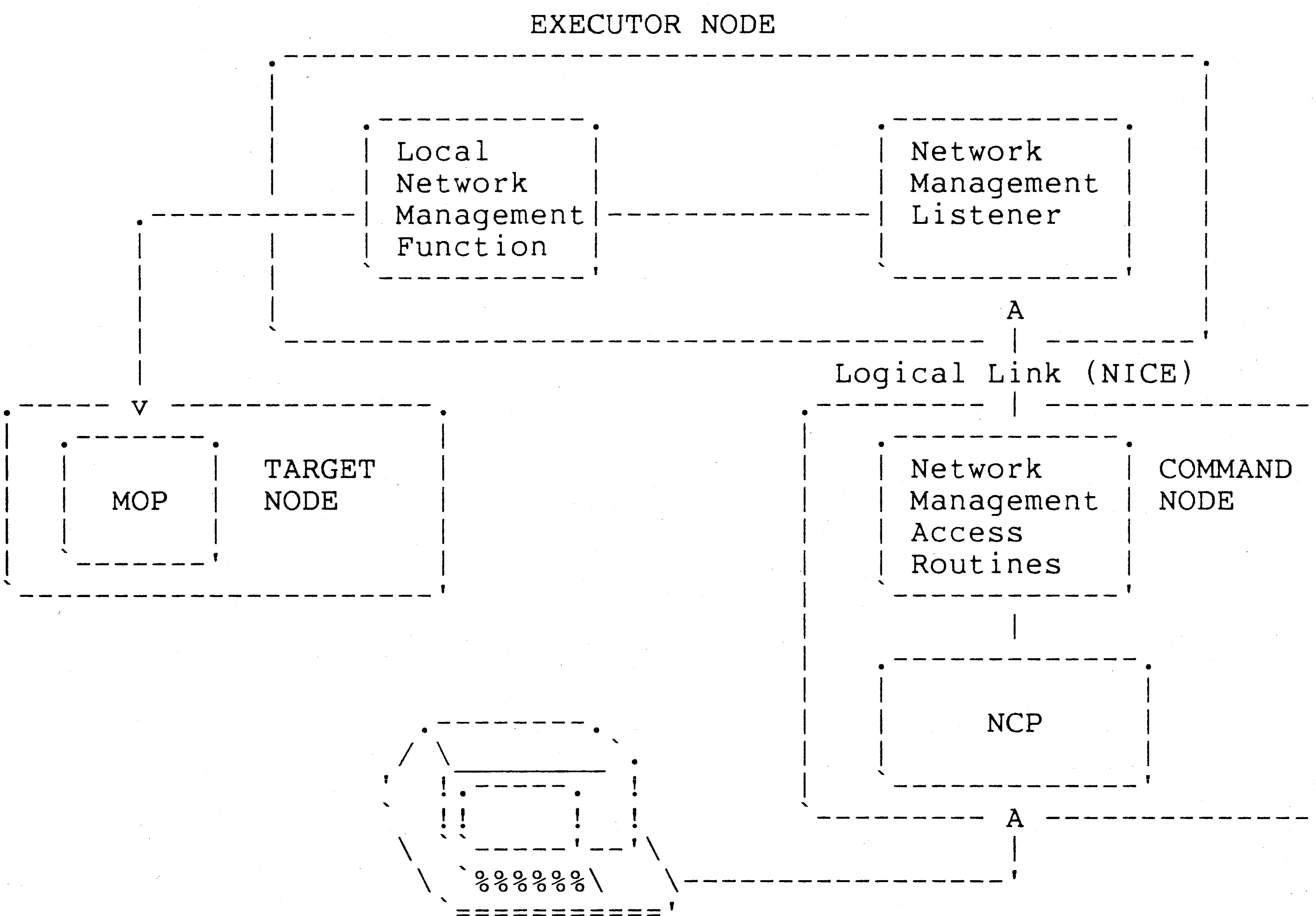


Figure 8B. Operator-initiated Down-Line Load Request Operation from a Remote Command Node

The executor proceeds with the load according to the options in the initial request.

Several fields in the NICE request down-line load message may be either furnished as overrides or defaulted to the values in the Dump/Load Server portion of the node data base. Any information left to default is first obtained from the data base.

The executor identifies the target node by address, name, or circuit. The name and address parameters may be supplied as overrides to those in the data bases. The address or link identification key into the node data base. If link is used, then address is obtained from the data base entry. If a target is identified by name, then address is determined by normal name to address mapping and used to key into the data base.

The address the target is to have is always sent to the target during the down-line load request operation. This target address is either obtained from the node data base or supplied as an override.

The name the target is to have, if any, is either supplied with the request as an override or obtained by normal address-to-name mapping.

Host identification follows similar rules to target identification. The host node address must be sent to the target. If both name and address are not supplied, address is obtained from the node data base. Name, if any, is obtained by normal address-to-name mapping, if not supplied.

The executor controls the process of loading the requested programs until the operating system is loaded. The executor is responsible for understanding the service protocol (for example, MOP) from and to the target.

The first program to run in the target node, called the primary loader, is typically loaded directly from its own bootstrap ROM. It then requests, over the communications line, the next program in the sequence. This program, the secondary loader, may have certain restrictions on the way it is loaded, depending on the capabilities of the primary loader. This process may extend through a tertiary loader. The final program to be loaded is defined as the operating system, although it does not necessarily have to be capable of being a network node. Within a single down-line process (possibly including "loader loads") each program loaded is expected to request another, except for the operating system, which does not.

When the down-line load has been completed (in other words, the operating system successfully loaded) or aborted due to an error, the executor sends the proper response back to the command node to finish up the process.

The content of the load image file is specified in Appendix E.

The algorithm for handling the down-line load is as follows:

Call Data Link Service Function to open link for load.
Call Maintenance Functions to perform load.
Call Data Link Service Function to close line.

5.7 Up-line Dump Operation

The up-line dump capability of the Network Management layer allows a system to dump its memory to a file on a network node.

The requirements for such a dump correspond with those for a down-line load:

- o The system being dumped must be connected to a network node (executor) by a specific physical link.
- o The system being dumped must run a minimal cooperative program that can communicate over the link with the executor. The protocol used is in the Low Level Maintenance Operation specification.
- o If the executor determines that the program is not there, then executor must supply the program. This is the secondary dumper.
- o The link used must be in the ON or SERVICE state and returned afterwards to its original state.
- o The executor must have access to the file receiving the dump. If the file is remote, the executor transfers the data using remote file access routines. (Refer to the DAP Functional Specification.)

The system to be dumped can indicate that it is capable of being dumped. In this case, the Link Watcher at the executor node senses the possibility of a dump and can pass a dump request to the Local Network Management Functions at the executor node. Alternatively, the executor or a remote command node can initiate the dump with an NCP DUMP command. In this case, the executor node's Local Network Management Functions receive the request from the Network Management Access Routines or the Network Management Listener.

The Local Network Management Functions proceed according to the options in the request. Any required information that has been left to default is first obtained from the node data base. The Local Network Management Functions then accomplish the dump using the Maintenance Functions and the local operating system's file system or network remote file transfer facilities. If the remote system does not respond, the executor can trigger the remote system and load a secondary dumping program.

In cases where the dump was not initiated by the target node, when the requested memory has been dumped to a file or the dump has been aborted, the executor sends an appropriate response back to the node requesting the operation.

The content of the dump file is specified in Appendix E.

The algorithm for performing the up-line dump is as follows:

Call Data Link Service Function to open line for dump.
Call Maintenance Functions to perform dump.
Call Data Link Service Function to close line.

5.8 Trigger Bootstrap Operation

The trigger bootstrap capability of the Network Management layer allows remote control of an operating system's restart capability. Since a system being booted is not necessarily a fully functional network node, the operation must be performed over a specific physical link. The node on the network side of the link is called the executor node.

The NCP TRIGGER command can initiate the trigger bootstrap function via the Network Management Listener and/or the Network Management Access Routines. The Local Network Management Functions at the executor node receive the request.

When the Local Network Management Functions receive a NICE trigger bootstrap request, they proceed according to the options in the request. Any required information which has been left to default is obtained from the node data base.

The physical link being used must be in the ON or SERVICE state at the executor node's end. The executor uses the Maintenance Functions to perform the operation.

When the operation is complete, the executor sends its response to the command node.

Once the target node is triggered, it will then load itself in whatever manner its bootstrap ROM is programmed to operate. This could include requesting a down-line load either from the executor that just triggered it or some other. The target node could load itself from its own mass storage.

The algorithm for implementing the trigger bootstrap is as follows:

Call Data Link Service function to open link for trigger.
Call Maintenance Functions to perform trigger.
Call Data Link Service function to close link.

5.9 Loop Test Operation

There are two types of loop tests, node level and data link level. Both types are loopback tests that loop a standard test block a specified number of times.

If either test fails, the response explains the failure. If the test fails because the test message was too long, the error return is "invalid parameter value, length" (Appendix F) and the test data field of the error message contains the maximum length of the loop test data, exclusive of test data overhead. If the test fails for any other reason, the test data field contains the number of messages that had not been looped when the test was declared a failure.

The unlooped count need not be returned for success or for errors that occur before looping can begin (for example, connect errors, command message format, or content errors). The only exception to this is the case that the value of the length parameter is too large, since this requires a return of the maximum length.

If the test is on an Ethernet circuit and no physical address is specified, the test is done with the loopback assistance multicast address. The responding station's physical address is included in the response.

5.9.1 Node Level Testing

There are two general categories of node level tests (shown in Figures 9 and 10, following). Both use normal traffic that requires logical links. Both have variations that use the Loopback Mirror and NCP LOOP NODE commands. The difference is that the first type uses what might be called "normal" communication, while the second type sets up a loop node name established with the NCP SET NODE CIRCUIT command.

The four ways in which node level messages travel are:

1. Local to local

2. Local to remote

3. Local to local loopback (using an operator-controlled loopback device with a loop node defined with the circuit to be used)

4. Local to remote loopback (using two connected nodes with a loop node defined with the circuit to be used)

The first two ways are used for the "normal" communication tests. The last two ways are used for the loop node name tests.

Test data can be a Loopback Mirror test message that is repeated a defined number of times, a file that is transferred in any of the ways listed above, or a message generated by a user task.

The set up commands for various types of node level tests are described in Figures 9A through 10D.

The operation of node level testing that uses Network Management modules is as follows. The Local Network Management Functions receive the NCP LOOP NODE command from the Network Management Listener and/or Network Management Access Routines. If a circuit is involved in the test, it must be in the ON state. If the Loopback Mirror is involved, the message is passed to the Loopback Mirror Access Routines (see Section 5.13). One logical link loop test uses a loop node with a routing node on the remote end of the line (Figures 9A - 9D). This test returns the test data on the circuit chosen by the Routing algorithm at the routing node.

```
SET LINE line-id CONTROLLER LOOPBACK
SET NODE FISHY CIRCUIT circuit-id
(Transfer file from FISHY)
```

NODE BOB

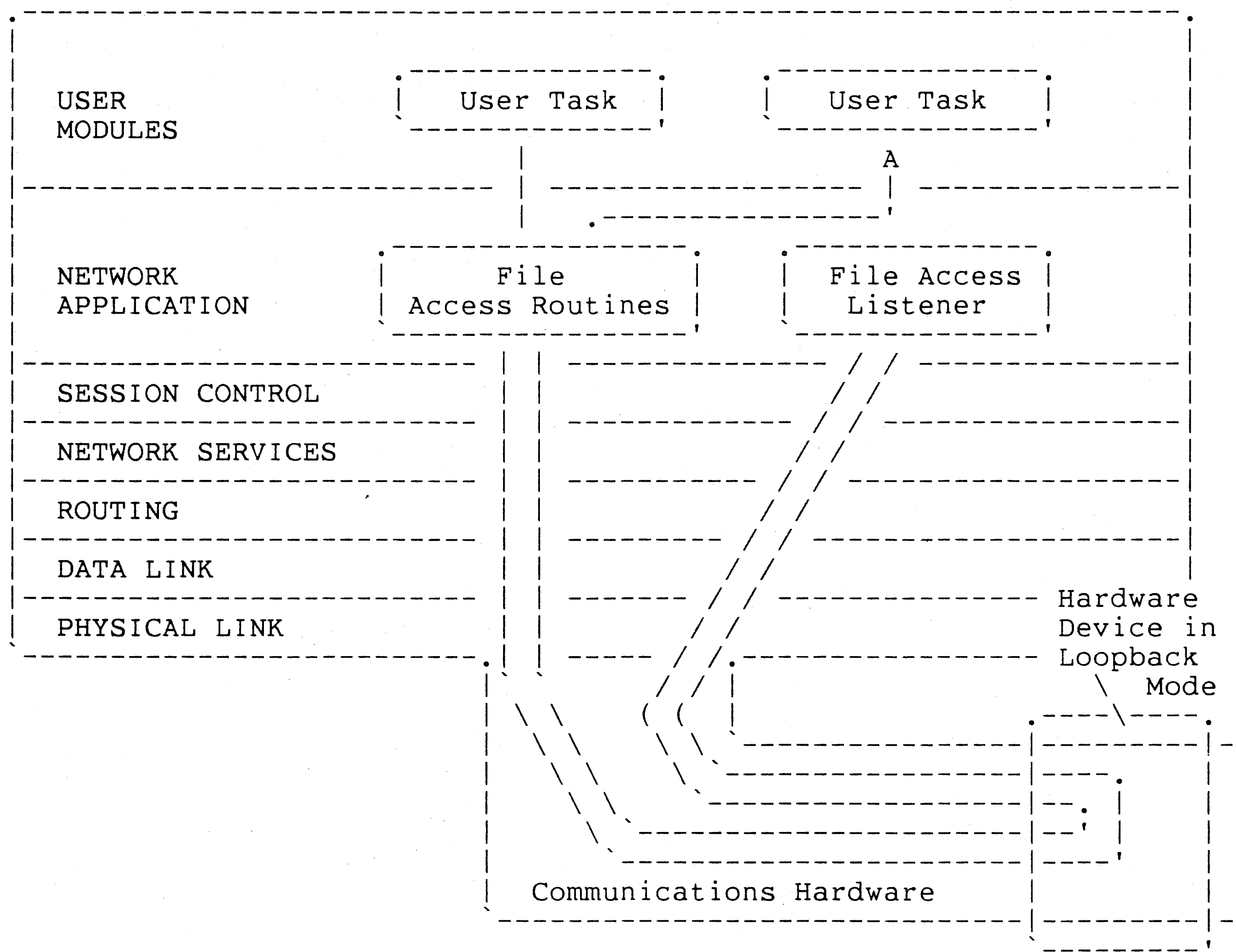


Figure 9A. Local-to-Loopback Node Test, Single Node (using files as test data, with a software controlled loopback capability)

```
SET NODE FISHY CIRCUIT circuit-id
LOOP NODE FISHY
```

NODE BOB

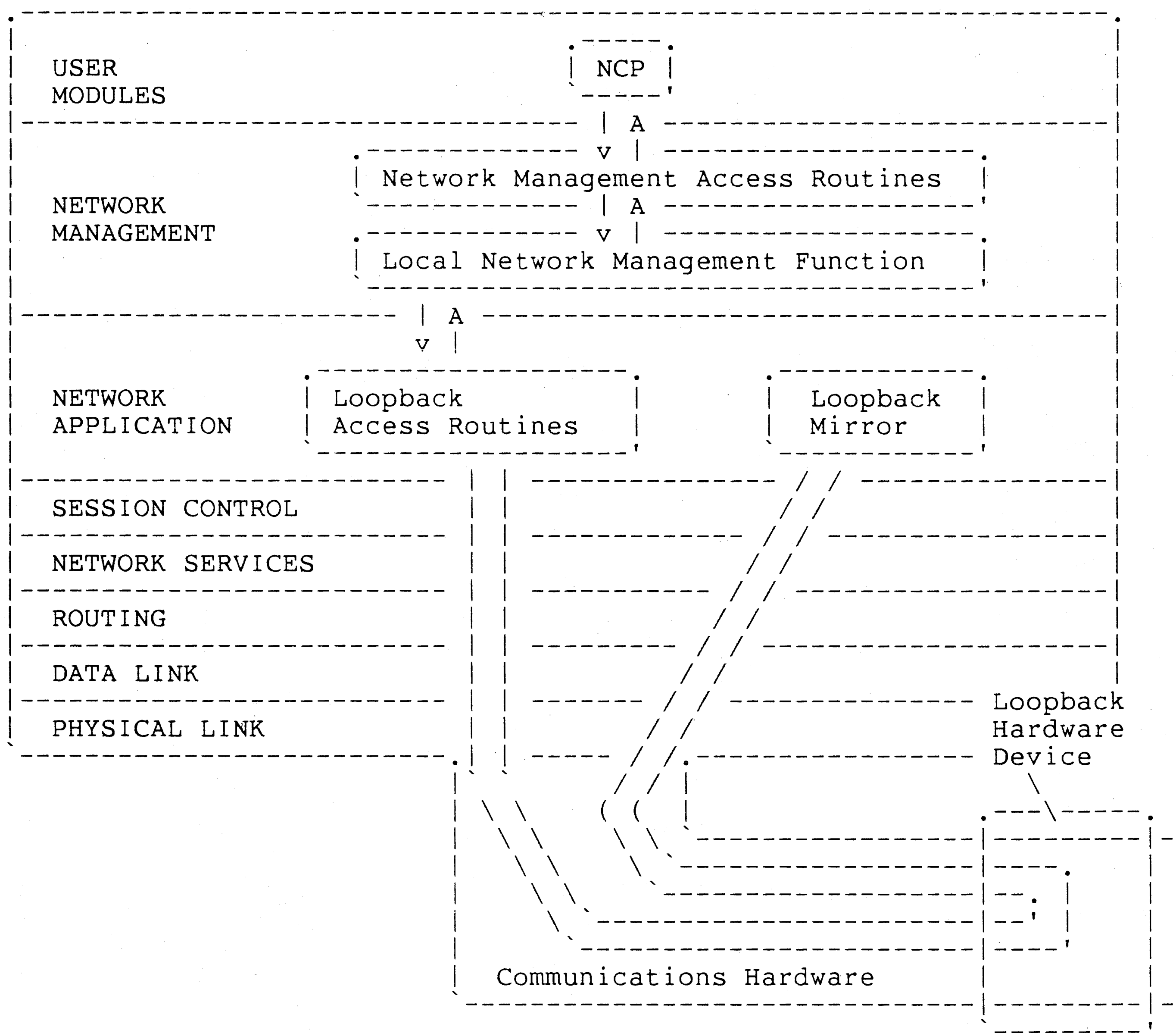


Figure 9B. Node Test, Single Node (using loopback mirror and test messages, and a manually set loopback device)

SET NODE FISHY CIRCUIT circuit-id
(Invoke user task using BOB and FISHY)

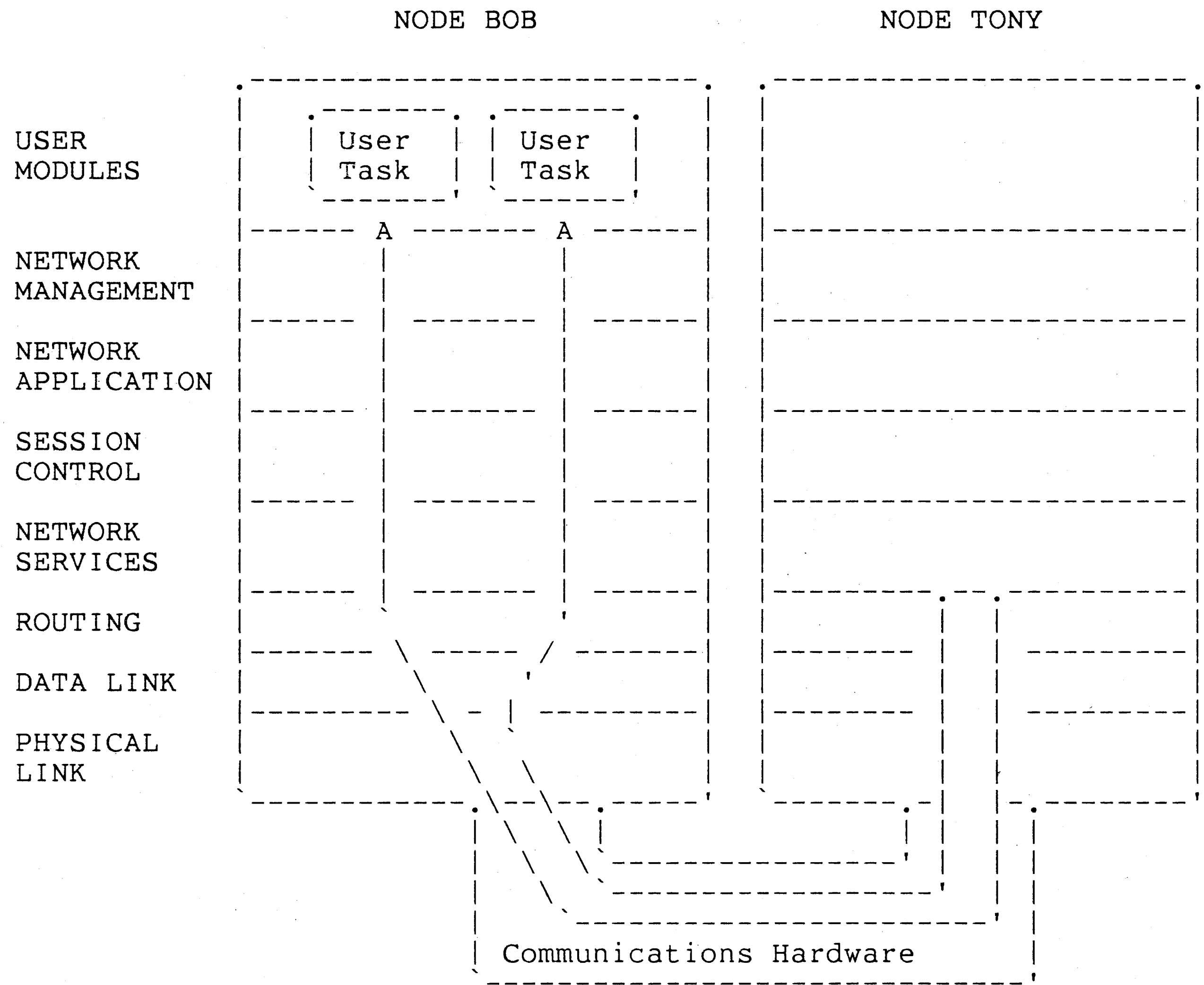


Figure 9C. Local-to-Loopback Node Test, Two Nodes (using user task)

SET NODE FISHY CIRCUIT circuit-id
LOOP NODE FISHY

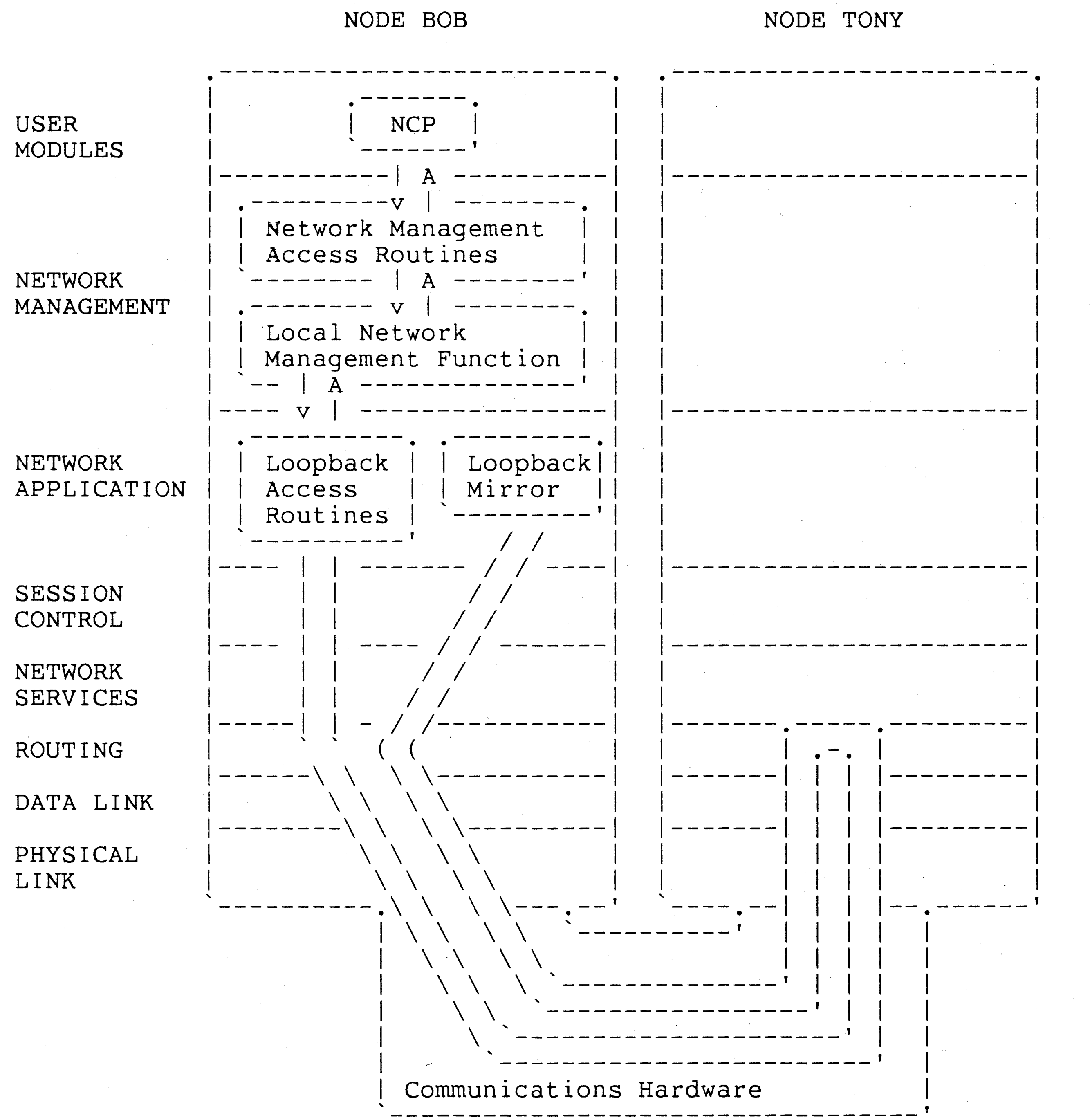


Figure 9D. Local-to-Loopback Node Test, Two Nodes (using loopback mirror and text messages)

(LOOP NODE BOB) or (LOOP EXECUTOR)

```
                              NODE BOB

 .-----------------------------------------------------------------------.
 |                                    .----.                             |
 |  USER                              | NCP |                            |
 |  MODULES                           `----'                             |
 |--------------------------------- | A -------------------------------|
 |                         .--------- v | ------------------.           |
 |                         | Network Management Access Routines |       |
 |  NETWORK                `--------- | A ------------------'           |
 |  MANAGEMENT             .--------- v | ------------------.           |
 |                         | Local Network Management Function |        |
 |                         `---------------------------------'          |
 |------------------------ | A ----------------------------------------|
 |                         v |                                          |
 |                     .------------------.       .-----------.         |
 |  NETWORK            | Loopback         |       |  Loopback |         |
 |  APPLICATION        | Access Routines  |       |  Mirror   |         |
 |                     `------------------'       `-----------'         |
 |-------------------------- | | ---------------- | | -----------------|
 |  SESSION CONTROL          | |                  | |                  |
 |-------------------------- | | ---------------- | | -----------------|
 |  NETWORK SERVICES         | |                  | |                  |
 |-------------------------- | | ---------------- | | -----------------|
 |  ROUTING                  | `------------------' |                  |
 |-------------------------- `--------------------' ------------------|
 |  DATA LINK                                                           |
 |----------------------------------------------------------------------|
 |  PHYSICAL LINK                                                       |
 `--------------------------.------------.-----------------------------'
                            |            |
                            |            |
                            |            `-----------------------------
                            |
                            |
                            |
                            |
                            |  Communications Hardware
                            `-----------------------------------------
```

Figure 10A. Normal Local-to-Local (using loopback mirror)

(Invoke user task using BOB)

NODE BOB

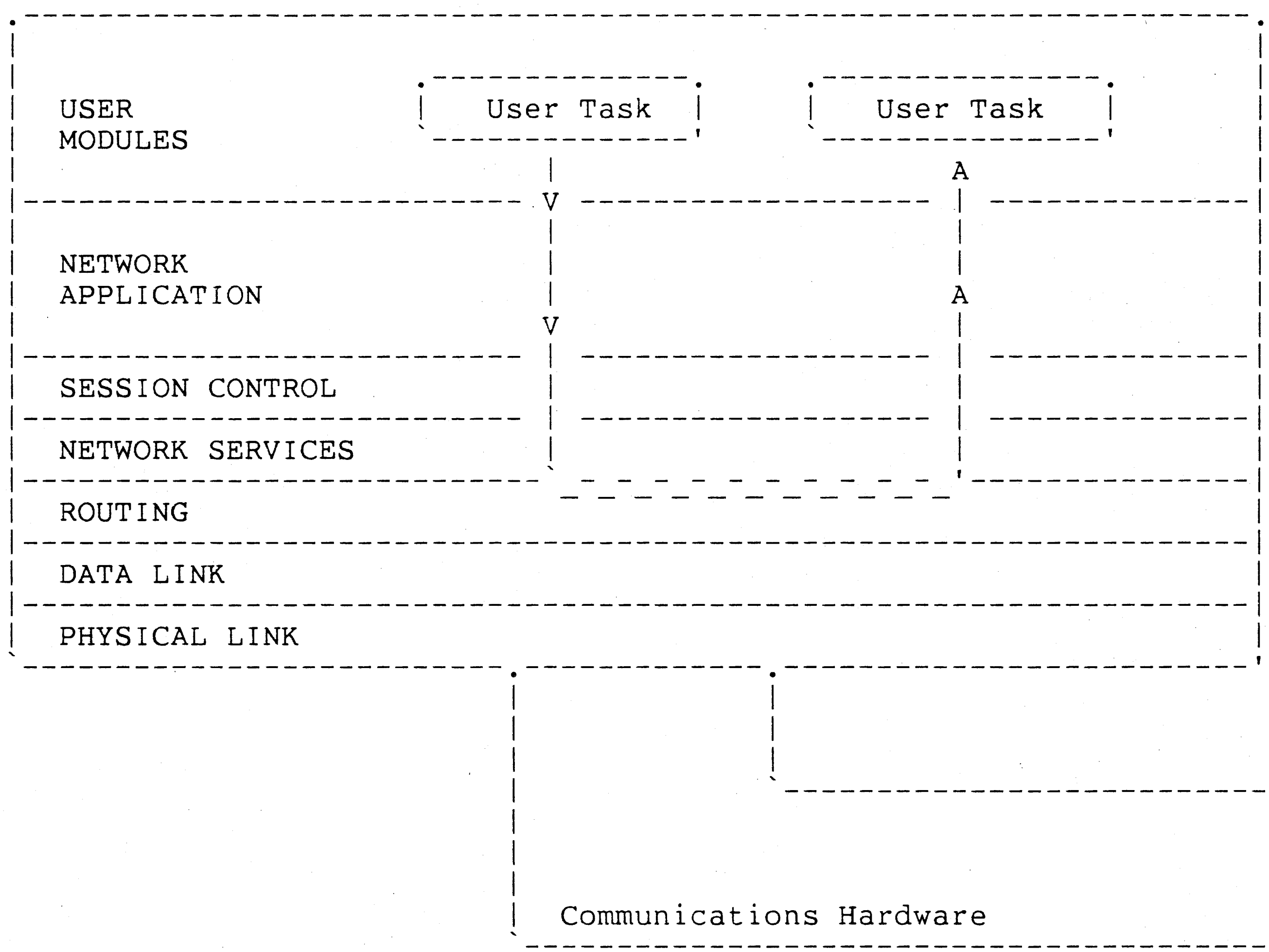


Figure 10B. Normal Local-to-Local (using user tasks)

LOOP NODE TONY

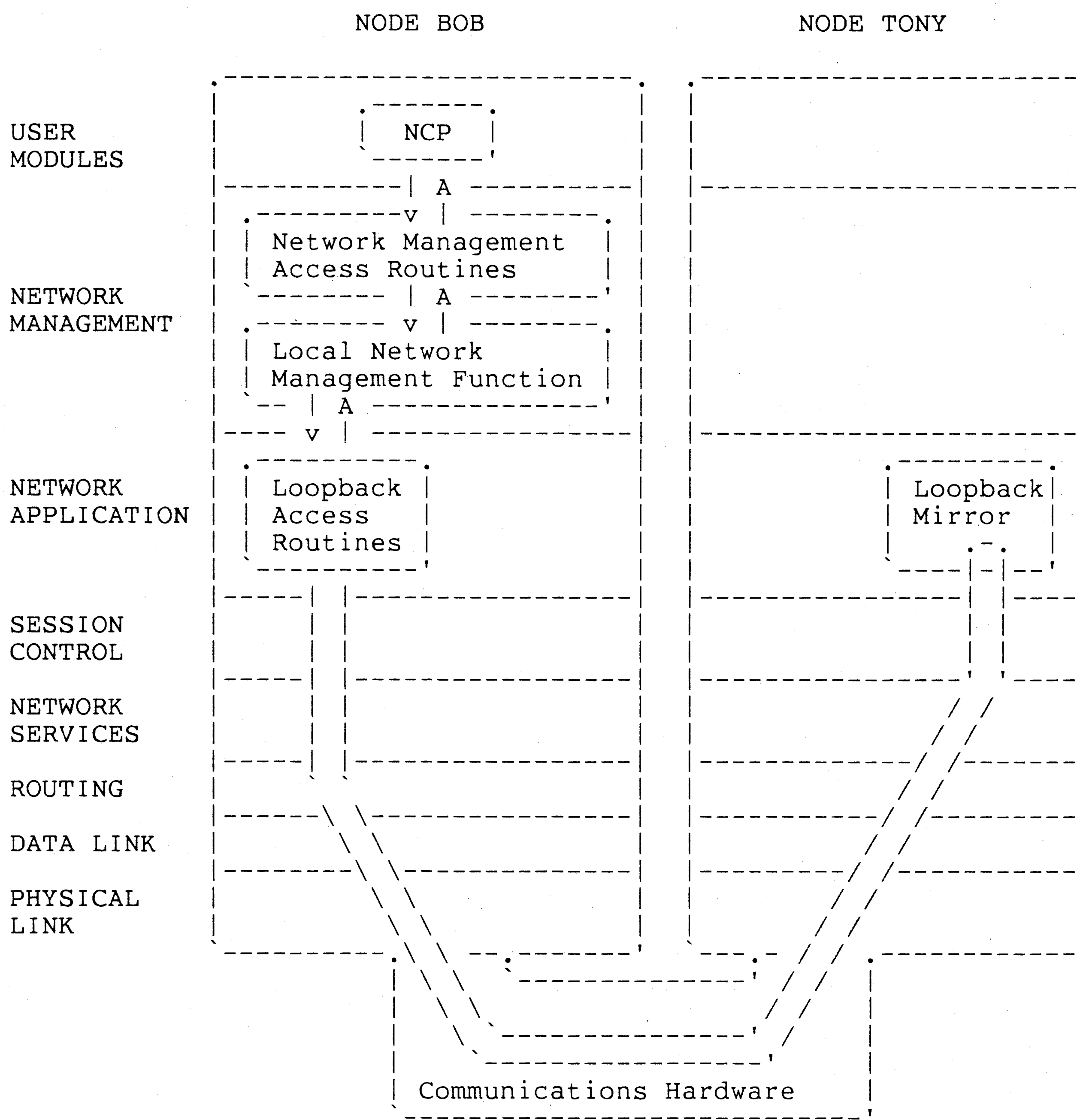


Figure 10C. Normal Local-to-Remote (using loopback mirror)

(Transfer files from BOB to TONY)

```
                          NODE BOB                     NODE TONY

                 .--------------------------.  .-------------------------.
                 |                          |  |                         |
 USER            |         .-------.        |  |        .-------.        |
 MODULES         |         | User  |        |  |        | User  |        |
                 |         | Task  |        |  |        | Task  |        |
                 |         `-------'        |  |        `-------'        |
                 |----------- | ------------|  |---------- A -----------|
                 |      .---- v ----.       |  |     .---- | ----.       |
 NETWORK         |      | File Access |     |  |     | File Access |     |
 APPLICATION     |      | Routines    |     |  |     | Listener    |     |
                 |      `-------------'     |  |     `-------------'     |
                 |----------- | ------------|  |---------- | ------------|
 SESSION         |            |             |  |           |             |
 CONTROL         |            v             |  |           A             |
                 |----------- | ------------|  |---------- | ------------|
 NETWORK         |            |             |  |           |             |
 SERVICES        |            |             |  |           |             |
                 |----------- | ------------|  |---------- | ------------|
 ROUTING         |            |             |  |           |             |
                 |----------- | ------------|  |---------- | ------------|
 DATA LINK       |            |             |  |           |             |
                 |----------- | ------------|  |---------- | ------------|
 PHYSICAL        |            |             |  |           |             |
 LINK            |            |             |  |           |             |
                 `--------.-- | --.---------'  `-------.-- | --.---------'
                          |   |   |                    |   |   |
                          |   |   |                    |   |   |
                          |   |   `--------------------'   |   |
                          |   |                            |   |
                          |   `----------------------------'   |
                          |                                    |
                          |   Communications Hardware          |
                          `------------------------------------'
```

Figure 10D. Normal Local-to-Remote (using files as test data)

5.9.2 Data Link Testing

Data Link level testing requires a direct interface between the Data Link Service Function and the Data Link layer. Figures 11A and 11B, at the end of this section, show two types of data link level tests:

1. Direct link loopback, hardware looped

2. Direct link loopback, software looped

Link loopback requires the use of data link service software (for example, MOP), with the link to be tested in the ON or SERVICE state.

The hardware-looped option requires an operator-controlled loopback controller, a modem set to loopback mode, a ROM with loopback capabilities at the remote end, or some other equivalent operation. It is recommended that the operator turn off the link, reconfigure the hardware, and then turn the link back on. Alternatively, the operator may leave the link in the ON state, and any resulting synchronization problem will be logged as an error.

The algorithm for the active loop test is as follows:

```
Set not done
Call Data Link Service Functions to open link for active loop
WHILE not done
     Call Maintenance Functions to loop message
     Call Data Link Service Function to receive message
     IF error OR count exhausted OR message is not loop data or
     looped data OR received data does not match sent data
          Set done
     ENDIF
ENDWHILE
Call Data Link Service Function to close link
```

```
{ [SET CIRCUIT circuit-id STATE OFF]          }
{ (manually set loopback device)              }
{ SET CIRCUIT circuit-id STATE ON/SERVICE *   }
{ LOOP CIRCUIT circuit-id                     }

                   OR

{ (circuit in ON or SERVICE state)            }
{ SET LINE line-id CONTROLLER LOOPBACK        }
{ (LOOP CIRCUIT circuit-id)                   }
```

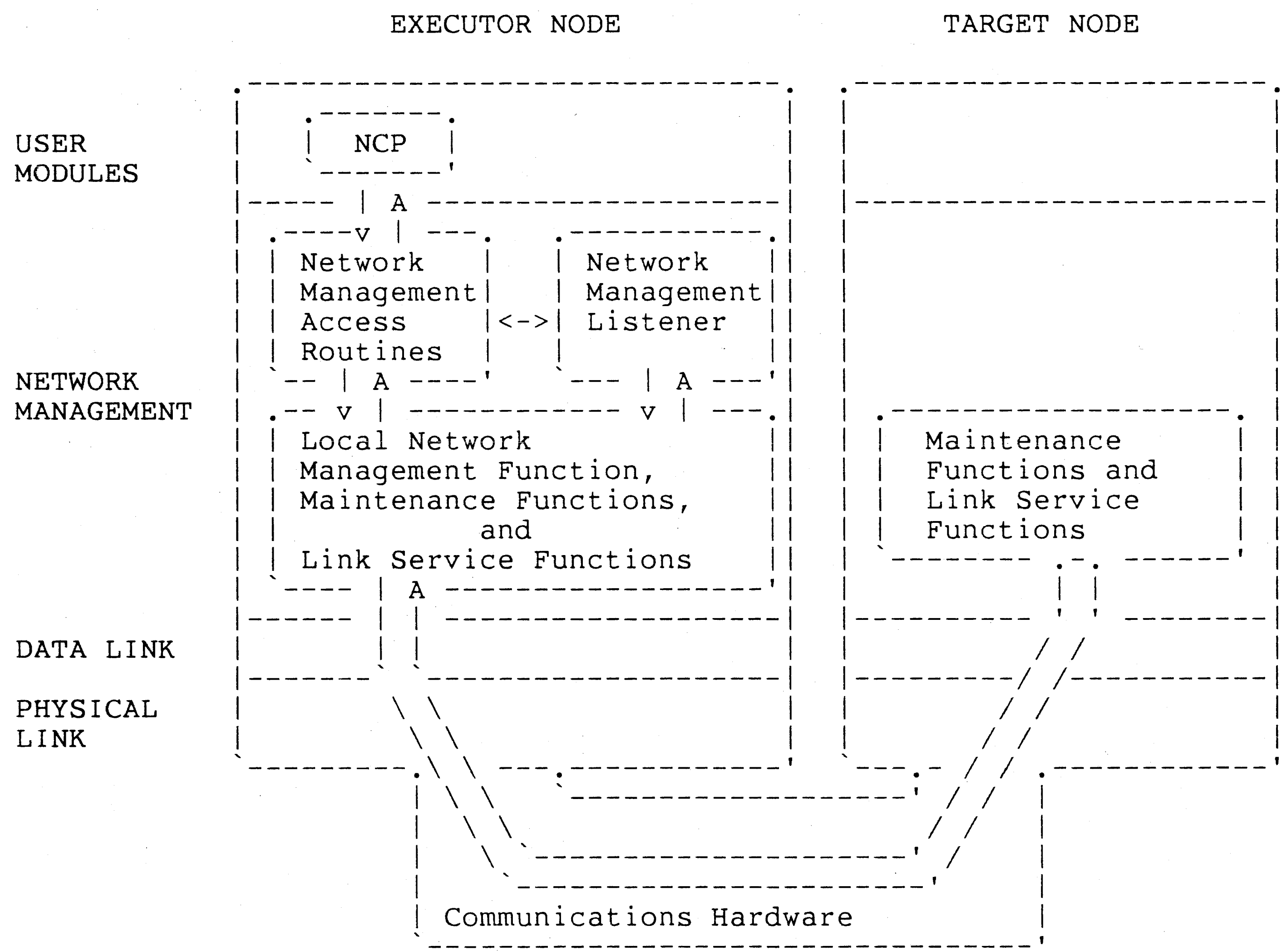
```
                          EXECUTOR NODE

 .-------------------------------------------------------------------.
 |                                .-----.                              |
 |  USER                          | NCP |                              |
 |  MODULES                       `-----'                              |
 |---------------------------- | A ----------------------------------|
 |                   .-------------- v | --.     .---------------------.  |
 |                   | Network Management |     | Network Management |  |
 |                   | Access Routines    |<---->| Listener           |  |
 |  NETWORK          `-------------- | A --'     `---------- | A ------'  |
 |  MANAGEMENT          .-------- v | ------------------- v | -.        |
 |                      | Local Network Management Functions, |        |
 |                      | Maintenance Functions, and          |        |
 |                      | Link Service Functions              |        |
 |                      `-------------------------------------'        |
 |-------------------------- |  A -----------------------------------|
 |  DATA LINK                |  |                                    |
 |-------------------------- |  | ---------------------------- Loopback
 |  PHYSICAL LINK            |  |                              Hardware
 `------------------------.  `  `  .-------------------------- Device
                          |           |                          \
                          |  \  \     |                    .-----\---.
                          |   \  \    `--------------------|---------|-
                          |    \  `------------------------|----.    |
                          |     `--------------------------|----'    |
                          |  Communications Hardware       |         |
                          `--------------------------------|---------|-
                                                           `---------'
```

* implementation dependent

Figure 11A. Direct Line, Hardware Looped, Data Link Loopback Tests and Command Sequences Effecting Them

LOOP CIRCUIT circuit-id
(circuit at TARGET NODE in SERVICE or ON state *)



* implementation dependent

Figure 11B. Direct Line, Software Looped, Data Link Loopback Tests and Command Sequences Effecting Them

5.10 Change Parameter Operation

When a NICE change parameter request is received, the specified parameters are changed, usually by interfacing with the local operating system. An appropriate response is then returned to the requester. The options of the change parameter request indicate the desired operation (either specifying a different value or removing the value) and the entity it relates to. The operation can be done either for volatile or permanent parameters.

The request may contain zero or more parameters. If there are none, the operation applies to the entire entity entry (in other words, the NCP ALL parameter). All parameters in the message should be checked before any are changed in the data base. If one parameter fails the check, then the operation should fail. A single response indicates success or failure for single-entity operations.

A change parameter request may apply to a group of entities. In this case, success or failure is individual. The entire request does not fail if a single entity request fails. An initial fail return implies no further responses are coming. A special success return indicates more responses will follow, one for each entity in the group.

Changing the link state requires the following capabilities:

For operator:

- o Set link state to OFF
- o Set link state to ON
- o Set link state to SERVICE

For the Link Watcher:

- o Set link state to ON-AUTOSERVICE
- o Reset link state from ON-AUTOSERVICE

All of the algorithms imply recording the link state if they succeed. The link state algorithms follow.

Set link state to OFF:

```
Call link's high level user to set link state to off
Call Line Service Function to set link state to off
```

Set link state to ON:

```
Call Data Link Service Function to set link state to passive
IF success
     Call link's high level user to set link state to on
ELSE
     Fail
ENDIF
```

Set link state to SERVICE:

```
Call Data Link Service Function to set link state to closed
IF success
     Call link's high level user to set link state to off
ELSE
     Fail
ENDIF
```

Set link state to ON-AUTOSERVICE:

```
IF link state is ON
    Perform algorithm to set link state to service
ELSE
    Fail
ENDIF
```

Reset link state from ON-AUTOSERVICE:

```
IF link state is ON-AUTOSERVICE
    Perform algorithm to set link state to on
ENDIF
```

5.11 Read Information Operation

When a read information request is received, a response is returned, followed by the requested data in the form of standard Network Management data blocks (Section 7). The data may be obtained either from within the Local Network Management Function itself or by interfacing with the system as appropriate.

The many restrictions and special situations relating to reading specific parameters or counters are described in Section 7. Additional information is in Section 4.3.8 (SHOW command).

A fail return in the first response implies no further responses are coming. A special success return indicates the command message was accepted and more will follow.

5.12 Zero Counters Operation

When a zero counters request is received, the appropriate counters are cleared by interfacing with the local operating system. An appropriate response is then returned to the requester.

If a read and zero was requested, the counters are returned as if a read information had been requested.

A fail return on the first response implies no further responses are coming. Success is a single return for single-entity operations. For multiple-entity operations, success is a special success return implying further responses.

5.13 Loopback Mirror Operation

The Loopback Mirror service tests logical links either between nodes or within a single node. It consists of an access interface -- the Loopback Access Routine; service routines -- the Loopback Mirror; and a simple protocol -- the Logical Loopback Protocol. The loopback mirror function operates in the Network Application DNA layer.

When the Loopback Mirror accepts a connect, it returns its maximum data size in the accept data. This is the amount of data it can handle, not counting the function code.

When a Logical Loopback message is received, it is changed into the appropriate response message and returned to the user (Figure 10, Section 5). The Loopback Mirror continues to repeat all traffic offered. The initiator of the link disconnects it.

5.14 NICE Logical Link Handling

This section describes the logical link algorithms that Network Management uses when sending NICE messages. The version data formats are in Section 6.12. The determination that a received version number is acceptable is always the responsibility of the higher version software, whether it is the command source or the listener.

The buffer size for NICE messages is 300 bytes.

The Network Management Listener algorithm follows:

```
Receive connect request
(Optionally) Determine privilege level based on access control
IF resources available and received version number OK
    Send connect accept with version number in accept data
    WHILE connected (see Note, below)
        Receive command message
        IF message received
            Process command message according to command and
                    privilege
            Send response message(s)
        ENDIF
    ENDWHILE
ELSE
    IF received version number not OK
        Send connect reject with version skew reason in reject
                data
    ELSE
        Send connect reject
    ENDIF
ENDIF
```

NOTE

> The algorithms used for connections are implementation dependent. For example, connections can be maintained permanently, only while the executor is set, timed-out, or one per command.

The Network Management command source algorithm follows:

```
Send connect request with version number in connect data
IF connect accepted
    IF received version number OK
        WHILE desired
            Send command message
            Receive response message(s)
        ENDWHILE
    ELSE
        Failure due to version skew
    ENDIF
    Disconnect link
ELSE
    IF connect rejected by listener
        IF reject data indicates version skew
            Failure due to version skew
        ELSE
            Failure due to listener resources
        ENDIF
    ELSE
        Failure due to network connect problem
    ENDIF
ENDIF
```

Use the following algorithm for an event transmitter:

```
Send connect request with version number in connect data
IF connect accepted
    IF received version number OK
        WHILE desired
            Send event message
        ENDWHILE
    ELSE
        Failure due to version skew
    ENDIF
    Disconnect link
ELSE
    Perform implementation specific error handling
ENDIF
```

Use the following algorithm for an event receiver:

```
Receive connect request
IF resources available and received version number OK
    Send connect accept with version number in accept data
    WHILE connected
        Receive event messages
    ENDWHILE
ELSE
    Send connect reject
ENDIF
```

5.15 Algorithm for Accepting Version Numbers

A version number consists of three parts -- version, ECO (Engineering Change Order), and user ECO (Section 6.12). In general, another version is acceptable if it is greater than or equal to this version. If less than this version, it is optionally acceptable as determined by product requirements.

When comparing two version numbers, compare the second parts only if the first parts are equal, and so on.

For Event Logging, the lack of a version number implies Version 2.0.

5.16 Return Code Handling

Use the following return code handling algorithm to call the Network Management access routines:

```
Initiate function
IF return code = more (2)
    WHILE return code <> done (-128)
        Perform next operation
        Process success/failure (1,3,<0)
    ENDWHILE
ELSE
    Process success/failure (1,<0)
ENDIF
```

Note that an initiate call starts the function, and an operate call performs the function (one entity at a time in the case of plural entities).

Use the following algorithm for deciding return codes within the Network Management access routines:

```
IF multiple returns needed
    Return "more" (2)
ENDIF
WHILE more returns
    IF success
        IF all response data for entity in single return
        OR last of multiple responses for this entity
            Return "success" (1)
        ELSE
            Return "partial" (3)
        ENDIF
    ELSE
        Return error code and other error information
    ENDIF
ENDWHILE
IF multiple returns needed
    Return "done" (-128)
ENDIF
```

Example:

The following sequence of messages might be returned in response to a SHOW ACTIVE CIRCUITS command.

```
(2)
UNA-0  1(RED) xxx  (3)
UNA-0  2(BLUE) yyy (3)
UNA-0  3(PINK) zzz (1)
UNA-1  91  (3)
UNA-1  92  (3)
UNA-1  93  (1)
UNA-2  (1)
DMC-2  (1)
(-128)
```

6 NETWORK MANAGEMENT MESSAGES

This section describes the NICE and Event Logging Messages, the NICE response message format, the NICE connect and accept data format, and the Logical Link Message format.

NICE is a command-response protocol. Because the Network Management layer is built on top of the End Communication and Data Link layers, which provide logical links that guarantee sequential and error-free data delivery, NICE does not have to handle error recovery.

In the message descriptions that follow, any unused bits or bytes are to be reserved and set to zero to allow compatibility with future implementations. Conditions such as non-zero reserved areas and unrecognized codes or unused bytes at the end of a field or message should be treated as errors, and no operation should be performed other than an appropriate error response.

The entire message should be parsed and checked for validity before any operation is performed.

The method for indicating that a function should be executed on all parameters in the data base for a particular entity (NCP ALL option) is to not include any parameters in the NICE function request message.

Parameters in command and response messages must be in ascending order by parameter type number, except that qualifiers must preceed the parameters they qualify. A parameter of the same type may be repeated for parameters that compose a list.

Qualifiers are restricted to one of the same parameter type per command or response message. If qualified parameters appear in a message the qualifier must appear in the same message. Unqualified parameters may or may not be included in a message with a qualifier. All qualified parameters in a message with a qualifier must be associated with that qualifier. In a sequence of multiple return messages, when qualifiers are nested, the outer level qualifier is not repeated until it changes.

6.1 NICE Function Codes

The NICE protocol performs the following message functions. The last one is for system specific commands, not specified in this document.

Function Code	NICE Function
15	Request down-line load
16	Request up-line dump
17	Trigger bootstrap
18	Test
19	Change parameter
20	Read information
21	Zero counters
22	System-specific function

6.2 Message Format Notation

The Network Management message format descriptions use the following notation.

FIELD (LENGTH) : CODING = Description of field

where:

FIELD

Is the name of the field being described

LENGTH

Is the length of the field as:

1. A number meaning number of 8-bit bytes.

2. A number followed by a "B" meaning number of bits.

3. The letters "EX-n" meaning extensible field with n being a number meaning the maximum length in 8-bit bytes. If no number is specified the length is limited only by the maximum NICE message. Extensible fields are variable in length consisting of 8-bit bytes, where the high-order bit of each byte denotes whether the next byte is part of the same field. The -1 means the next byte is part of this field while a 0 denotes the last byte. Extensible fields can be binary or bit map; if binary, then 7 bits from each byte are concatenated into a single binary field; if bit map, then 7 bits from each byte are used independently as information bits. The bit definitions define the information bits after removing extension bits and compressing the bytes.

4. The letters "I-n" meaning image field with n being a number which is the maximum length in 8-bit bytes of the image. The image is preceded by a 1-byte count of the length of the remainder of the field. Image fields are variable length and may be null (count-0). All 8 bits of each byte are used as information bits. The meaning

and interpretation of each image field is defined with that specific field.

5. The character "*" meaning remainder of message. A number following the asterisk indicates the minimum field length in bytes.

CODING

Is the representation type used.

where:

A = 7-bit ASCII

B = Binary

BM = Bit Map (where each bit or group of bits has independent meaning)

C = Constant

Notes:

1. If length and coding are omitted, FIELD represents a generic field with a number of subfields specified in the descriptions.

2. Any bit or field which is stated to be "reserved" shall be zero unless otherwise specified. Any bit or field not described is reserved.

3. All numeric values in this document are shown in decimal representation unless otherwise noted.

4. All fields are presented to the physical link protocol least significant byte first. In an ASCII field, the leftmost character is in the low-order byte.

5. Bytes in this document are numbered with bit 0 the rightmost (low-order, least-significant) bit, and bit 7 the leftmost (high-order, most-significant) bit. Fields and bytes of other lengths are numbered similarly.

6. Corresponding data type format notation used in Section 7 is described at the beginning of that section.

6.3 Request Down-line Load Message Format

FUNCTION CODE	OPTION	NODE	CIRCUIT	PARAMETER ENTRIES

where:

FUNCTION CODE (1) : B = 15

OPTION (1) BM Is one of the following options:

Option bit	Value/Meaning
0-2	0 = Identify target by node-id. 3 = Identify target by circuit-id.

NODE Is the target node identification (see Section 7) as key into defaults data base (present only if option = 0). Plural nodes options are not allowed.

CIRCUIT Is the circuit identification (see Section 7). Plural circuits options not allowed. Present only if option = 3.

PARAMETER ENTRIES are zero or more of PARAMETER ENTRY consisting of:

DATA ID	DATA

where:

DATA ID (2) : B Is the parameter type number (see note below and Section 7).

DATA Is the parameter data (see Section 7).

NOTE

The parameters allowed are the following node parameters:

ADDRESS
CPU
DIAGNOSTIC FILE
HOST
LOAD FILE
NAME
PHYSICAL ADDRESS
SECONDARY LOADER
SERVICE DEVICE
SERVICE CIRCUIT (allowed only if option = 0)
SERVICE PASSWORD
SOFTWARE IDENTIFICATION
SOFTWARE TYPE
TERTIARY LOADER

6.4 Request Up-line Dump Message Format

```
FUNCTION  OPTION  NODE  CIRCUIT   PARAMETER
CODE                               ENTRIES
```

where:

FUNCTION CODE (1): B = 16

OPTION (1) : BM — Is one of the following options:

Option bits	Value/Meaning
0-2	0 = Identify target by node-id. 3 = Identify target by circuit-id.

NODE — Identifies the node to be dumped (present only if option = 0). Format is defined in Section 7.10.

CIRCUIT — Specifies the circuit over which to dump (present only if option = 3). Format is defined in Section 7.4.

PARAMETER ENTRIES — Are zero or more of PARAMETER ENTRY consisting of:

```
DATA    DATA
 ID
```

where:

DATA ID (2) : B — Is the parameter type number (see note below and Section 7).

DATA — Is the parameter data (see Section 7).

NOTE

The parameters are selected from the node parameters. Only certain parameters are allowed in the dump message. They are:

DUMP ADDRESS
DUMP COUNT
DUMP FILE
PHYSICAL ADDRESS
SECONDARY DUMPER
SERVICE CIRCUIT (allowed only if option = 0)
SERVICE PASSWORD

6.5 Trigger Bootstrap Message Format

```
FUNCTION   OPTION  NODE   CIRCUIT    PARAMETER
  CODE                                ENTRIES
```

where:

FUNCTION CODE (1): B = 17

OPTION (1) : BM Is one of the following options:

Option bits	Value/Meaning
0-2	0 = Identify target by node-id. 3 = Identify target by circuit-id.

NODE — Identifies the node to trigger boot on (present only if option = 0). The format is defined in Section 7.10.

CIRCUIT — Identifies the circuit over which to trigger the boot (present only if option = 3). The format is defined in Section 7.4.

PARAMETER ENTRIES — Are zero or more of PARAMETER ENTRY consisting of:

```
DATA    DATA
 ID
```

where:

DATA ID (2) : B — Is the parameter type number (see note below and Section 7).

DATA — Is the parameter data (see Section 7).

NOTE

The parameters are selected from the node parameters. Only certain parameters are allowed in the trigger message. They are:

PHYSICAL ADDRESS
SERVICE CIRCUIT (allowed only if option = 0)
SERVICE PASSWORD

6.6 Test Message Format

FUNCTION CODE	OPTION	NODE	USER	PASSWORD	ACCOUNTING	LINK	PARAMETER ENTRIES

where:

FUNCTION CODE (1): B = 18

OPTION (1) : BM Is one of the following options:

Option bits	Value/Meaning
0-2	0 = Node type loop test 1 = Line loop test 3 = Circuit loop test

If node type loop test:

7	0 = Default access control 1 = Access control included

For node type loop tests only (option 0), four parameters are as follows:

NODE — Identifies the node to loopback the test block in node-id format. Plural node options are not allowed.

USER (I-39): A — Is the user-id to use when connecting to node. Present only if option bit 7 = 1.

PASSWORD (I-39): A — Is the password to use when connecting to node. Present only if option bit 7 = 1.

ACCOUNTING (I-39):A — Is the accounting information to use when connecting to node. Present only if option bit 7 = 1.

For line or circuit tests only (option 1 or 3), one parameter is as follows:

LINK — Identifies the link to send the test on in circuit- or line-id format. Plural options not allowed.

PARAMETER ENTRIES — Are zero or more of PARAMETER ENTRY, consisting of:

DATA ID	DATA

where:

DATA ID (2) : B Is the parameter type number (see note below and Section 7).

DATA — Is the parameter data (see Section 7).

NOTE

The parameters are selected from the node parameters. Only certain parameters are allowed in the test message. They are:

LOOP ASSISTANT NODE
LOOP ASSISTANT PHYSICAL ADDRESS
LOOP COUNT
LOOP HELP
LOOP LENGTH
LOOP NODE
LOOP WITH
PHYSICAL ADDRESS

6.7 Change Parameter Message Format

FUNCTION CODE	OPTION	ENTITY ID	PARAMETER ENTRIES

where:

FUNCTION CODE (1): B = 19

OPTION (1): BM — Is one of the following options:

Bits	Meaning
7	0 = Change volatile parameters.
	1 = Change permanent parameters.
6	0 = Set/define parameters.
	1 = Clear/purge parameters.
0-2	Entity type (Section 7).

ENTITY ID — Identifies the particular entity (Section 7).

PARAMETER ENTRIES — Are zero or more of PARAMETER ENTRY consisting of:

DATA ID	DATA

where:

DATA ID (2) :	B	Is the parameter type number (see Section 7).
DATA		Is the parameter data (see Section 7).

NOTE

The DATA field is not present when option bit 6 is set unless the parameter is a qualifier rather than a parameter that is to be cleared.

6.8 Read Information Message Format

FUNCTION CODE	OPTION	ENTITY ID	PARAMETER ENTRIES

where:

FUNCTION CODE (1): B = 20

OPTION (1): BM Is one of the following options:

Bits	Meaning
7	0 = Read volatile parameter 1 = Read permanent parameter
4-6	Information type as follows: 0 = Summary 1 = Status 2 = Characteristics 3 = Counters 4 = Events
0-2	Entity type (Section 7).

ENTITY ID Identifies the particular entity (Section 7).

PARAMETER ENTRIES 0 or more parameter entries, formatted as for change parameter message. These are limited to the following qualifiers:

CIRCUIT ADJACENT NODE
LOGGING SINK NODE
MODULE X25-ACCESS NETWORK
MODULE X25-PROTOCOL DTE GROUP
MODULE X25-SERVER DESTINATION
NODE CIRCUIT

6.9 Zero Counters Message Format

FUNCTION CODE	OPTION	ENTITY ID	PARAMETER ENTRIES

where:

FUNCTION CODE (1): B = 21

OPTION (1): BM — Is one of the following options:

Bits	Meaning
7	1 = Read and zero 0 = Zero only
0-2	Entity type (Section 7). (Circuit, line, module, or node only).

ENTITY ID — Identifies the particular entity, if required (Section 7).

PARAMETER ENTRIES — 0 or more parameter entries, formatted as for change parameter message. These are limited to the following qualifiers:

MODULE X25-PROTOCOL DTE

6.10 NICE System Specific Message Format

FUNCTION CODE	SYSTEM TYPE	REMAINDER

where:

FUNCTION CODE (1) : B = 22

SYSTEM TYPE (1) : B — Represents the type of operating system command to which command is specific.

Value	System
1	RSTS
2	RSX family
3	TOPS-10/20
4	VMS
5	RT
6	CT
7	Communications Server

REMAINDER (*) : B — Consists of data, depending on system specific requirements.

6.11 NICE Response Message Format

RETURN CODE	ERROR DETAIL	ERROR MESSAGE	ENTITY ID	TEST DATA	DATA BLOCK

where:

RETURN CODE (1) : B — Is one of the standard NICE return codes (Appendix F).

[ERROR DETAIL] (2) : B — Is more detailed error information according to the error code (e.g., a parameter type). Zero if not applicable. If applicable but not available, its value is 65,535 (all bits set). In this case it is not printed. Applicable only if 0 > RETURN CODE > -128.

[ERROR MESSAGE] (I-255) : A — Is a system-dependent error message that may be output in addition to the standard error message.

[ENTITY ID] — Identifies a particular entity (Section 7) if operation is on plural entities, or operation is read information or read and zero counters. If the entity is the executor node, bit 7 of the name length is set.

[TEST DATA] (2) : B — Is the information resulting from a test operation (Test message only). This is only required if a test failed and if data is relevant, or a data block is present in this message. It is UNLOOPED COUNT or MAXIMUM LOOP DATA, depending on ERROR DETAIL. Section 5.9 further explains its contents.

[DATA BLOCK] — Is one of the data blocks described in Section 7 (returned for read information message or read and zero message). For Test messages it may contain the parameter PHYSICAL ADDRESS.

If a response message is short terminated after any field, the existing fields may still be interpreted according to standard format. This means, for example, that a single byte return is to be interpreted as a return code.

Responses to messages not noted as exceptions above are single responses indicating return code, error detail, and error message.

A success response to a request for plural entities is indicated by a return code of only 2 with no other fields, followed by a separate response message for each entity. Each of these messages contains the basic response data (return code, error detail, and error message) and the entity id. A return code of -128 indicates the end of multiple responses.

6.12 NICE Connect Initiate and Connect Accept Data Formats

The first three bytes of the connect initiate and connect accept data are:

VERSION	DEC ECO	USER ECO

where:

VERSION (1) : B Is the version number

DEC ECO (1) : B Is the DIGITAL ECO number

USER ECO (1) : B Is the user ECO number

6.13 Event Message Binary Data Format

This section describes the generalized binary format of event data. It applies to messages on logical links and, as much as possible, to files.

The buffer size for event messages is 200 bytes.

The format of an event logging message is:

FUNCTION CODE	SINK FLAGS	EVENT CODE	EVENT TIME	SOURCE NODE	EVENT ENTITY	EVENT DATA

where:

FUNCTION CODE (1) : B = 1, meaning event log

SINK FLAGS (1) : BM Are flags indicating which sinks are to receive a copy of this event, one bit per sink. The bit assignments are:

Bit	Sink
0	Console
1	File
2	Monitor

EVENT CODE (2) : BM Identifies the specific event as follows:

Bits	Meaning
0-4	Event type
6-14	Event class

EVENT TIME Is the source node date and time of event processing. Consists of:

JULIAN HALF DAY | SECOND | MILLISECOND

where:

JULIAN HALF DAY (2) :	B	Number of half days since 1 Jan 1977 and before 9 Nov 2021 (0-32767). For example, the morning of Jan 1, 1977 is 0.
SECOND (2) :	B	Second within current half day (0-43199).
MILLISECOND (2) :	B	Millisecond within current second (0-999). If not supported, high order bit is set, remainder are clear, and field is not printed when formatted for output.

SOURCE NODE Identifies the source node. It consists of:

NODE ADDRESS | NODE NAME

where:

NODE ADDRESS (2) :	B	Node address (see Section 7.9).
NODE NAME (I-6) :	A	Node name, 0 length, if none.

EVENT ENTITY Identifies the entity involved in the event, as applicable. Consists of:

ENTITY TYPE | ENTITY ID

where:

ENTITY TYPE (2) :	B	Represents the type of entity. A -1 value indicates no entity. A value >= 0 is the entity type and is followed by the entity id in its usual format.

EVENT DATA (*) : B Is event specific data, zero or more data entries as defined for NICE data blocks, parameter types according to event class.

6.14 Logical Loopback Message Formats

6.14.1 Connect Accept Data Format

MAXIMUM DATA

where:

MAXIMUM DATA (2) : B Is the maximum length, in bytes, that the Loopback Mirror can loop.

6.14.2 Command Message Format

FUNCTION DATA
CODE

where:

FUNCTION CODE (1) : B = 0

DATA (*) : B Is the data to loop.

6.14.3 Response Message Format

RETURN CODE DATA

where:

RETURN CODE (1) : B Indicates Success (1) or Failure (-1).

DATA (*) : B Is the data as received, if success.

7 PARAMETER AND COUNTER BINARY FORMATS AND VALUES

This section describes the binary formats of all entities, parameters, counters, and events, as well as the returns used in the NICE protocol and Event Logging messages in response to a request for information. Section 3 describes the entities, parameters, counters, and events along with their user level formats.

7.1 Introduction to Binary Format Descriptions

Read this section before the rest of Section 7. This section explains notation format, symbols, and other general information pertaining to all entities. Since the symbols and notation are only explained in this section, you may need to refer back to it when studying Tables 5 thru 29.

7.1.1 Type Numbers

Each entity, parameter and counter is assigned a type number. The entity type numbers are as follows:

Type Number	Keyword
0	NODE
1	LINE
2	LOGGING
3	CIRCUIT
4	MODULE
5	AREA

Entity type fields have 3-bit lengths.

The parameter and counter type numbers appear in the tables in this section.

7.1.2 Entity Parameter Identifier Format

The following format is for input (NCP to NML) of an entity type parameter:

ENTITY TYPE (1): B

Is the entity type. The values are defined as type numbers in section 7.1.1. If the entity type is NODE, it is followed by a node identification.

7.1.3 String Identifier Format

The string type identifiers use the following format. (Section 6.1 describes this format notation.)

ID FORMAT (1): B Is the identification format type, with the following values:

Number	Meaning
-5	Significant (if applicable)
-2	Active (if applicable)
-1	Known (if applicable)
>0	Length of identification

ID: A Is the ASCII identification if ID FORMAT >0.

7.1.4 Node Identifier Formats

When represented in binary, node identification is one of a number of different formats (limited by the particular function). All choices begin with a format type. This applies to all occurrences of node identification. The input (NCP to NML) format is as follows:

NODE FORMAT (1): B Represents the node format type, as follows:

Number	Type
-5	Significant nodes, no further data
-4	Adjacent nodes, no further data
-3	Loop nodes, no further data
-2	Active nodes, no further data
-1	Known nodes, no further data
0	Node address
>0	Length of node name, followed by the indicated number of ASCII characters.

In the ENTITY ID field of a response message bit 7 set indicates the node identification is the executor node.

NODE ADDRESS (2): B Is the node address if NODE FORMAT = 0. When used as input, a node address of zero implies the executor node.

NODE NAME: A Is the node name if NODE FORMAT >0.

The usual binary output (NML to NCP) format is as follows:

```
NODE    NODE
ADDRESS NAME
```

where:

NODE ADDRESS (2): B Is the node address. When supplied as output a node address of 0 indicates a loop node.

NODE NAME (I-6): A Is the node name, 0 length implies none.

7.1.5 Area Identifier Format

When represented in binary, the format of an area identification is as follows:

AREA FORMAT (1): B Represents the area format type, as follows:

Number	Type
-2	Active areas, no further data
-1	Known area, no further data
0	Area number

AREA NUMBER (1): B Is the area number if AREA FORMAT = 0.

7.1.6 Object Format for Entity Types

The following format is for input of an object parameter:

OBJECT FORMAT (1): B Is the object identification format type, with the following values:

Value	Meaning
0	Numeric object number
>0	Length of object name

If the OBJECT FORMAT = 0:

OBJECT NUMBER (1): B Is the object type number.

If the OBJECT FORMAT >0:

OBJECT NAME: A Is the ASCII object type.

On output the value is either an object number or an object name, distinguishable by the data type.

7.1.7 Numeric Range

All occurrences of a numeric range use a common format. On input, the format is:

BEGINNING (2):	B	Is the range beginning.
END (2):	B	Is the range end. If range consists of a single number, END equals BEGINNING.

The output format for a range is a coded multiple field with two numbers for a range and one number for a single value. For multiple (disjoint) ranges, the parameter must be repeated.

7.1.8 Parameter Display Format and Descriptive Encoding Notation

Each parameter is assigned a data type at Network Management layer level that describes the format of the parameter. This information allows NCP to format and output most parameter values in a simple way, even if NCP does not recognize the parameter type.

The notation used in the parameter tables in this section to describe these data types is as follows:

Notation	Data Type
C-n	Coded, single field, n bytes
CM-n	Coded, multiple field, n fields
NC	Not coded (any of the following)
AI-n	ASCII image field, maximum n bytes
DU-n	Decimal number, unsigned, n bytes
DS-n	Decimal number, signed, n bytes
H-n	Hexadecimal number, n bytes
HI-n	Hexadecimal image, maximum n bytes
O-n	Octal number, n bytes

Image formats (AI-n and HI-n) are displayed left to right in the order in which the bytes of data appear in the NICE message, i.e., the order of printed text. For Ethernet address or protocol type parameters, each byte of an HI-n data image (two characters) is separated from the next byte by a hyphen (-). Numbers are displayed most significant digits first with no separation of bytes. Hence, hexadecimal images appear in reverse byte order from hexadecimal numbers.

7.1.9 NICE Returns

A response to a SHOW command consists of the identification of the particular entity to which it applies and zero or more data entries. The data entries are either parameter or counter entries, depending on the information requested.

When an implementation recognizes the parameter type of a coded field, the value output should be the keyword(s) or other interpretation that corresponds to the code for that parameter type. If the parameter type is not recognized, the field should be formatted as hexadecimal.

The format of a data entry is as follows:

DATA ID (2): BM =

Identifies particular data entity:

Bit	Value	Meaning
15	0	Parameter data. The rest of the bits are as follows:

Bits	Meaning
0-11	Parameter type, interpreted according to entity type.
12-14	Reserved

Bit	Value	Meaning
15	1	Counter data. The rest of the bits are as follows:

Bits	Value	Meaning
0-11		Counter type
12	0	not bit mapped
	1	bit mapped
13-14	0	reserved
	1	8 bit counter
	2	16 bit counter
	3	32 bit counter

DATA TYPE (1): BM =

Identifies data type, present only for parameter data:

Bit	Value	Meaning
7	0	Not coded. The rest of the bits are as follows:

Bit	Value	Meaning
6	0	Binary number. The rest of the bits are as follows:

Bits	Value	Meaning
0-3	0	implies data is image field.
	>0	data length.
4-5	0	Unsigned Decimal Number
	1	Signed Decimal Number
	2	Hexadecimal Number
	3	Octal Number

6	1	ASCII image field. Bits 0-5 zero.

7 1 Coded, interpreted according to PARAMETER TYPE. The rest of the bits are as follows:

Bit	Value	Meaning
6	0	Single field. Bits 0-5 are the number of bytes in the field.
	1	Multiple field. Bits 0-5 are the number of fields, maximum 31; each field is preceded by a DATA TYPE.

BIT MAP (2): BM

Is the counter qualifier bit map, included only if DATA ID is counter and counter is bit mapped.

DATA: B

Is the data, according to data id and type.

The data required for setting a parameter or counter is the entity identification, the DATA ID, and the DATA. The information required for clearing a parameter or counter is the entity identification and the DATA ID. When a parameter is displayed, the information is entity id, DATA ID, DATA TYPE, BITMAP (if applicable) and DATA. The purpose of the data type field is to provide information for an output formatter. Thus the formatter can know how to format a parameter value even if its parameter type is unrecognized.

A coded multiple (CM) field cannot appear as a data type for a field within a coded multiple type parameter value.

All numbers are low byte first in binary form whether image or not. The image option for numbers can only be used for parameters where it is explicitly required. All number bases except hexadecimal have a maximum length of four bytes.

Indicate counter overflow by setting all bits in the DATA field.

The following ranges are reserved for system specific counters or parameters:

Range	Reserved for
2100-2299	RSTS specific
2300-2499	RSX specific
2500-2699	TOPS-10/20 specific
2700-2899	VMS specific
2900-3099	RT specific
3100-3299	CT specific
3300-3499	Communication Server specific
3500-3899	Future use

3900-4095 Customer specific

7.1.10 Information Types

Each parameter is associated with one or more information types. The parameter tables in this section use the following symbols to indicate information types for each parameter.

Symbol	Keyword	Associated Entity
C	CHARACTERISTICS	All entities
S	STATUS	All entities
*	SUMMARY	All entities
EV	EVENTS	LOGGING
Q		Qualifier

Qualifier indicates that the parameter is used as a qualifier for the parameters that follow it.

7.1.11 Applicability Restrictions

All node parameters and counters cannot be displayed at every node; nor can all line counters be displayed for every line-id. In the following tables, which describe the entity parameters and counters, the following symbols note these restrictions:

Symbol	Applicability
A	Adjacent node only
DN	Destination node only (includes executor)
E	Executor node only
N	Node by name only
L	Loop nodes
R	Remote nodes (all nodes except executor and loop nodes)
S	Sink node only
Q	Indicates a parameter must be qualified

7.1.12 Setability Restrictions

Some parameters have user setability restrictions, indicated in this section by the following notation:

Symbol	Meaning
RO	Read only

WO	Write only, in the sense that it appears in a different form in a read function. (For example, a node name can be set, but it is read as part of a node id.)
LO	Loop only

7.2 Circuit Parameters

The following table specifies the circuit parameters.

Table 5
Circuit Parameters

Type Number	Data Type	Inf. Type	App. Rest.	Set. Rest.	NCP Keywords
0	C-1	S*			STATE
1	C-1	S*		RO	SUBSTATE
100	C-1	C			SERVICE
110	DU-2	C			COUNTER TIMER
120	HI-6	S*,Q		RO	SERVICE PHYSICAL ADDRESS
121	C-1	S*	Q	RO	SERVICE SUBSTATE
200	CM-1/2	S*		RO	CONNECTED NODE
	DU-2				node address
	AI-6				node name (optional)
201	CM-1/2	S*		RO	CONNECTED OBJECT
	DU-1				object number
	AI-16				object name
400	AI-6	S*		RO	LOOPBACK NAME
800	CM-1/2	S*[1],Q		RO	ADJACENT NODE
	DU-2				node address
	AI-6				node name (optional)
801	CM-1/2	S		RO	DESIGNATED ROUTER
	DU-2				node address
	AI-6				node name (optional)
810	DU-2	S	Q	RO	BLOCK SIZE
811	DU-2	C			ORIGINATING QUEUE LIMIT
900	DU-1	C			COST
901	DU-1	C			MAXIMUM ROUTERS
902	DU-1	C			ROUTER PRIORITY
906	DU-2	C			HELLO TIMER
907	DU-2	C	Q	RO	LISTEN TIMER
910	C-1	C			BLOCKING
920	DU-1	C			MAXIMUM RECALLS
921	DU-2	C			RECALL TIMER
930	AI-16	C			NUMBER

1000	CM-2/3	S*	RO	USER
	C-1			Entity type
	AI-16			Entity name (if entity is not node)
	DU-2			Node address (if entity is node)
	AI-6			Node name (if entity is node)
1010	C-1	S*		POLLING STATE
1011	C-1	S*	RO	Polling substate
1100	CM-2/3	C		OWNER (Format same as for USER)
1110	AI-16	C		LINE
1111	C-1	C		USAGE
1112	C-1	C		TYPE
1120	AI-16	C		DTE
1121	DU-2	C		CHANNEL
1122	DU-2	C		MAXIMUM DATA
1123	DU-1	C		MAXIMUM WINDOW
1140	DU-1	C		TRIBUTARY
1141	DU-2	C		BABBLE TIMER
1142	DU-2	C		TRANSMIT TIMER
1145	C-1	C		MAXIMUM BUFFERS
1146	DU-1	C		MAXIMUM TRANSMITS
1150	DU-1	C		ACTIVE BASE
1151	DU-1	C		ACTIVE INCREMENT
1152	DU-1	C		INACTIVE BASE
1153	DU-1	C		INACTIVE INCREMENT
1154	DU-1	C		INACTIVE THRESHOLD
1155	DU-1	C		DYING BASE
1156	DU-1	C		DYING INCREMENT
1157	DU-1	C		DYING THRESHOLD
1158	DU-1	C		DEAD THRESHOLD

[1] SHOW CIRCUITS STATUS for broadcast circuits shows all adjacent nodes. SHOW CIRCUITS SUMMARY for broadcast circuits shows only router adjacent nodes.

The values for STATE are:

Value	Keyword
0	ON
1	OFF
2	SERVICE
3	CLEARED

The values for SUBSTATE and SERVICE SUBSTATE are:

Value	Keyword
0	STARTING
1	REFLECTING
2	LOOPING

3	LOADING
4	DUMPING
5	TRIGGERING
6	AUTOSERVICE
7	AUTOLOADING
8	AUTODUMPING
9	AUTOTRIGGERING
10	SYNCHRONIZING
11	FAILED

The values for SERVICE are:

Value	Keyword
0	ENABLED
1	DISABLED

The values for BLOCKING are:

Value	Meaning
0	ENABLED
1	DISABLED

The values for entity type and entity name can be found in section 7.1.1.

The values for POLLING STATE are:

Value	Keyword
0	AUTOMATIC
1	ACTIVE
2	INACTIVE
3	DYING
4	DEAD

The values for polling substate are:

Value	Keyword
1	ACTIVE
2	INACTIVE
3	DYING
4	DEAD

The values for USAGE are:

Value	Meaning
0	PERMANENT
1	INCOMING
2	OUTGOING

The values for TYPE are:

Value	Meaning
0	DDCMP POINT
1	DDCMP CONTROL
2	DDCMP TRIBUTARY
3	X25
4	DDCMP DMC
5	
6	Ethernet
7	CI
8	QP2 (DTE20)
9	BISYNC

The values for MAXIMUM BUFFERS are:

Range	Meaning
1-254	Number of buffers
255	UNLIMITED

7.3 Circuit Counters

The circuit entity counters are listed in Tables 6-8, following. The definition of each counter and the way that it is incremented can be found in the functional specification for the appropriate layer. Due to hardware characteristics, some devices cannot support all counters. In general, those counters that make sense are supported for all devices. Specific exceptions related to the DMC are noted in Appendix I.

Circuit counters are specified for the following layers only:

Layer	Type Number Range
Network Management	000 - 099
Routing	800 - 899
Data Link	1000 - 1999

The following counters are kept for all circuits, with the exception of type number 805, which is X.25 only.

Table 6
Circuit Counters Kept for All Circuits

Type Number	Bit Width	Standard Text
0	16	Seconds Since Last Zeroed
800	32	Terminating Packets Received
801	32	Originating Packets Sent
802	16	Terminating Congestion Loss
805	8	Corruption Loss
810	32	Transit Packets Received
811	32	Transit Packets Sent
812	16	Transit Congestion Loss
820	8	Circuit Down
821	8	Initialization Failure

The following Data Link counters apply to DDCMP circuits:

Table 7
Data Link Circuit Counters for DDCMP Circuits

Type Number	Bit Width	Standard Text	Bit Number Standard Text
1000	32	Bytes Received	
1001	32	Bytes Sent	
1010	32	Data Blocks Received	
1011	32	Data Blocks Sent	
1020	8	Data Errors Inbound	1 NAKs Sent, Data Field Block Check error 2 NAKs Sent, REP Response
1021	8	Data Errors Outbound	0 NAKs Received, Header Block Check error 1 NAKs Received, Data Field Block Check error 2 NAKs Received, REP Response
1030	8	Remote Reply Timeouts	
1031	8	Local Reply Timeouts	
1040	8	Remote Buffer Errors	0 NAKs Received Buffer Unavailable 1 NAKs Received Buffer Too Small
1041	8	Local Buffer Errors	0 NAKs Sent Buffer Unavailable 1 NAKs Sent Buffer Too Small
1050	16	Selection Intervals Elapsed	
1051	8	Selection Timeouts	0 No Reply to Select 1 Incomplete Reply to Select

The following Data Link counters apply to permanent X.25 circuits:

Table 8
Data Link Circuit Counters for Permanent X.25 Circuits

Type Number	Bit Width	Standard Text
1000	32	Bytes received
1001	32	Bytes sent
1010	32	Data blocks received
1011	32	Data blocks sent
1240	8	Locally initiated resets
1241	8	Remotely initiated resets
1242	8	Network initiated resets

The following table specifies Data Link counters for Ethernet circuits:

Table 9
Data Link Circuit Counters for Ethernet Circuits

Type Number	Bit Width	Standard Text
0	16	Seconds Since Last Zeroed
1000	32	Bytes Received
1001	32	Bytes Sent
1010	32	Data Blocks Received
1011	32	Data Blocks Sent
1065	16	User buffer unavailable

7.4 Line Parameters

The following table specifies the line parameters:

Table 10
Line Parameters

Type Number	Data Type	Inf. Type	Set. Rest.	NCP Keywords
0	C-1	S*		STATE
1	C-1	S*	RO	substate (not a keyword)
100	C-1	C		SERVICE
110	DU-2	C		COUNTER TIMER
1100	AI-16	C		DEVICE
1105	DU-2	C		RECEIVE BUFFERS
1110	C-1	C		CONTROLLER

1111	C-1	C		DUPLEX
1112	C-1	C		PROTOCOL
1113	C-1	C		CLOCK
1120	DU-2	C		SERVICE TIMER
1121	DU-2	C		RETRANSMIT TIMER
1122	DU-2	C		HOLDBACK TIMER
1130	DU-2	C		MAXIMUM BLOCK
1131	DU-1	C		MAXIMUM RETRANSMITS
1132	DU-1	C		MAXIMUM WINDOW
1150	DU-2	C		SCHEDULING TIMER
1151	DU-2	C		DEAD TIMER
1152	DU-2	C		DELAY TIMER
1153	DU-2	C		STREAM TIMER
1160	HI-6	C	RO	HARDWARE ADDRESS

The values for STATE, substate, and SERVICE are as listed in Section 7.2 for circuit parameters.

Communication DEVICE mnemonics (names) are:

Value	Name	Device
0	DP	DP11-DA (OBSOLETE)
1	UNA	DEUNA multiaccess communication link
2	DU	DU11-DA synchronous line interface
3	CNA	
4	DL	DL11-C, -E or -WA asynchronous line interface
5	QNA	
6	DQ	DQ11-DA (OBSOLETE)
7	CI	Computer Interconnect interface
8	DA	DA11-B or -AL UNIBUS link
9	PCL	PCL11-B multiple CPU link
10	DUP	DUP11-DA synchronous line interface
12	DMC	DMC11-DA/AR, -FA/AR, -MA/AL or -MD/AL interprocessor link
14	DN	DN11-BA or -AA automatic calling unit
16	DLV	DLV11-E, -F, -J, MXV11-A or - B asynchronous line interface
18	DMP	DMP11 multipoint interprocessor link
20	DTE	DTE20 PDP-11 to KL10 interface
22	DV	DV11-AA/BA synchronous line multiplexer
24	DZ	DZ11-A, -B, -C, or -D asynchronous line multiplexer
28	KDP	KMC11/DUP11-DA synchronous line multiplexer
30	KDZ	KMC11/DZ11-A, -B, -C, or -D asynchronous line multiplexer
32	KL	KL8-J (OBSOLETE)
34	DMV	DMV11 interprocessor link
36	DPV	DPV11 synchronous line interface
38	DMF	DMF-32 synchronous line unit
40	DMR	DMR11-AA, -AB, -AC, or -AE interprocessor link
42	KMY	KMS11-PX synchronous line interface with X.25 level 2 microcode
44	KMX	KMS11-BD/BE synchronous line interface with X.25

level 2 microcode

The values for PROTOCOL are:

Value	Meaning
0	DDCMP POINT
1	DDCMP CONTROL
2	DDCMP TRIBUTARY
3	(reserved)
4	DDCMP DMC
5	LAPB
6	Ethernet
7	CI
8	QP2 (DTE20)

The values for DUPLEX are:

Value	Keyword
0	FULL
1	HALF

The values for CONTROLLER are:

Value	Keyword
0	NORMAL
1	LOOPBACK

The values for CLOCK are:

Value	Meaning
0	EXTERNAL
1	INTERNAL

7.5 Line Counters

The following table specifies the Data Link counters for LAPB lines.

Table 11
Data Link Line Counters for LAPB Lines

Type Number	Bit Width	Standard Text	Bit Number Standard Text
0	16	Seconds Since Last Zeroed	
1000	32	Bytes Received	
1001	32	Bytes Sent	
1010	32	Data Blocks Received	
1011	32	Data Blocks Sent	
1020	8	Data Errors Inbound	3 Block too long 4 Block check error 5 REJ sent
1021	8	Data Errors Outbound	3 REJ received
1030	8	Remote Reply Timeouts	
1031	8	Local Reply Timeouts	
1040	8	Remote Buffer Errors	2 RNR received, buffer unavailable
1041	8	Local Buffer Errors	2 RNR sent, buffer unavailable
1100	8	Remote Station Errors	4 Invalid N(R) received 5 FRMR sent, header format error
1101	8	Local Station Errors	2 Transmit underrun 4 Receive overrun 5 FRMR received, head format error

The following table specifies Data Link counters for DDCMP lines:

Table 13
Data Link Line Counters for DDCMP Lines

Type Number	Bit Width	Standard Text	Bit Number Standard Text
1020	8	Data Errors Inbound	0 NAKs sent, header block check error
1100	8	Remote Station Errors	0 NAKs received, receive overrun 1 NAKs sent, header format error 2 Selection address errors 3 Streaming tributaries
1101	8	Local Station Errors	0 NAKs sent, receive overrun 1 Receive overruns, NAK not sent 2 Transmit underruns 3 NAKs received, header format error

The following table specifies Data Link counters for Ethernet lines:

Table 13.1
Data Link Line Counters for Ethernet Lines

Type Number	Bit Width	Standard Text	Bit Number Standard Text
0	16	Seconds Since Last Zeroed	
1000	32	Bytes Received	
1001	32	Bytes Sent	
1002	32	Multicast Bytes Received	
1010	32	Data Blocks Received	
1011	32	Data Blocks Sent	
1012	32	Multicast Blocks Received	
1013	32	Blocks sent, initially deferred	
1014	32	Blocks sent, single collision	
1015	32	Blocks sent, multiple collisions	
1060	16	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	16	Collision detect check failure	
1062	16	Receive failure	0 Block check error
			1 Framing error
			2 Frame too long
1063	16	Unrecognized frame destination	
1064	16	Data overrun	
1065	16	System buffer unavailable	
1066	16	User buffer unavailable	

7.6 Logging Parameters

When represented in binary, sink type is:

SINK TYPE (1): B   Represents the logging sink type as follows:

Value	Meaning
-2	Active sink types
-1	Known sink types
1	CONSOLE
2	FILE
3	MONITOR

Sections 7.11 and 7.12 define all the event classes and their associated events and parameters (not to be confused with the logging parameters).

Line and node counters provide information for event logging. There are no logging entity counters specified, just status, characteristics, and events.

The following table specifies the logging parameters:

Table 14
Logging Parameters

Param.	NICE Data Type	Info Type	Appl Restr.	NCP Keywords
0	C-1	S*	E	STATE
100	AI-255	C*	E	NAME
200	CM-1/2	EV*	S	SINK NODE
	DU-2			Node address
	AI-6			Node name (optional)
201	CM-2/3/4/5	EV*	S	EVENTS
	C-1			Entity type
	DU-2			Node address (if entity type is node)
	AI-6			Node name (if entity type is node) (optional)
	AI-16			Entity id (if entity type is not node)
	C-2			Event class
	HI-8			Event mask (if single event class indicated)

The values for STATE are:

Value	Keyword
0	ON
1	OFF
2	HOLD

The values for entity type are:

Value	Meaning
-1	No entity
0	NODE
1	LINE
3	CIRCUIT
4	MODULE
5	AREA

The event class specification is:

Bits	Meaning
14-15	0 = Single class 2 = All events for class 3 = KNOWN EVENTS
0-8	Event class if bits 14-15 equal 0 or 2.

The event mask specification is:

Event mask, bits set to correspond to event types (Table 22, Section 7.11). Low order bytes first. High order bytes not present imply 0 value. Format for NCP input or output is a list of numbers corresponding to the bits set (Section 3.2). Only present if EVENT CLASS is for a single class (bits 14-15 = 0).

NOTE

The wild card and KNOWN EVENTS specifications are for changing events only. Return read events as a class and mask.

7.7 Module Parameters

7.7.1 Console Module Parameters

The following table specifies the maintenance console module parameters.

Table 14a
Console Module Parameters

Type Number	Data Type	Inf. Type	App. Rest.	Set. Rest.	NCP Keywords
110	DU-2	C*			RESERVATION TIMER

7.7.2 Loader Module Parameters

The following table specifies the maintenance loader module parameters.

Table 14b
Loader Module Parameters

Type Number	Data Type	Inf. Type	App. Rest.	Set. Rest.	NCP Keywords
10	C-1	S*			ASSISTANCE

The values for ASSISTANCE are:

Value	Meaning
0	ENABLED
1	DISABLED

7.7.3 Looper Module Parameters

The following table specifies the maintenance looper module parameters.

Table 14c
Looper Module Parameters

Type Number	Data Type	Inf. Type	App. Rest.	Set. Rest.	NCP Keywords
10	C-1	S*			ASSISTANCE

The values for ASSISTANCE are:

Value	Meaning
0	ENABLED
1	DISABLED

7.7.4 Configurator Module Parameters

The following table specifies the maintenance configurator module parameters.

Table 14d
Configurator Module Parameters

Type Number	Data Type	Inf. Type	App. Rest.	Set. Rest.	NCP Keywords
100	AI-16	Q			CIRCUIT
110	C-1	S*	Q		SURVEILLANCE
111	CM-3	S*	Q	RO	ELAPSED TIME
	DU-2				Hours
	DU-1				Minutes
	DU-1				Seconds
120	HI-6	S,Q	Q	RO	PHYSICAL ADDRESS
130	CM-5	S	Q	RO	LAST REPORT
	DU-1				Day
	C-1				Month
	DU-1				Hour
	DU-1				Minute
	DU-1				Second
1001	CM-3	S	Q	RO	MAINTENANCE VERSION
	DU-1				Version number
	DU-1				ECO number
	DU-1				User ECO number
1002	CM-1-16	S	Q	RO	FUNCTIONS
	C-1				1-16 functions
1003	HI-6	S	Q	RO	CONSOLE USER
1004	DU-2	S	Q	RO	RESERVATION TIMER
1005	DU-2	S	Q	RO	COMMAND SIZE
1006	DU-2	S	Q	RO	RESPONSE SIZE
1007	HI-6	S	Q	RO	HARDWARE ADDRESS
1100	C-1	S	Q	RO	DEVICE
1200	CM-1/2	S	Q	RO	SOFTWARE IDENTIFICATION
	C-1				Generic Software type
	AI-16				Software ID string (present only if generic software type > 0)
1300	C-1	S	Q	RO	SYSTEM PROCESSOR
1400	C-1	S	Q	RO	DATA LINK
1401	DU-2	S	Q	RO	DATA LINK BUFFER SIZE

The values for SURVEILLANCE are:

Value	Meaning
0	ENABLED
1	DISABLED

The values for SYSTEM PROCESSOR and DATA LINK are found in the DNA Low Level Maintenance Operation specification. The values for DEVICE are the same as for the node parameter SERVICE DEVICE. The format and values of SOFTWARE IDENTIFICATION are found in the DNA Low level Maintenance Operation specification.

NOTE

The parameter type values for the information taken from the DNA Low Level Maintenance Operation System Identification message are the parameter types from that message plus 1000.

The values for FUNCTIONS are the corresponding bit numbers for the functions in the similar fields in the System Identification message.

The values for LAST REPORT month are 1-12 corresponding to month keywords JAN through DEC, respectively.

7.7.5 X.25 Access Module Parameters

The following table specifies the X.25 access module parameters.

Table 15
X.25 Access Module Parameters

Type Number	Data Type	Inf. Type	App. Rest.	Set. Rest.	NCP Keywords
320	CM-1/2	C	Q		NODE
	DU-2				node address
	AI-6				node name (optional if none)
330	AI-39	C	Q		USER
331	AI-39	C	Q	WO	PASSWORD (to set)
331	C-1	C	Q	RO	PASSWORD (to read)
332	AI-39	C	Q		ACCOUNT
1110	AI-16	Q			NETWORK

On output, the PASSWORD parameter is not present if there is no password. If it is present, its value is:

Value	Meaning
0	Password set

7.7.6 X.25 Protocol Module Parameters

The following table specifies the X.25 protocol module parameters.

Table 16
X.25 Protocol Module Parameters

Type Number	Data Type	Inf. Type	App. Rest.	Set. Rest.	NCP Keywords
0	C-1	S*	Q		STATE
1	C-1	S*		RO	Substate (not a keyword)
100	DU-2	C	Q		COUNTER TIMER
1000	DU-2	S*	Q	RO	ACTIVE CHANNELS
1010	DU-2	S*	Q	RO	ACTIVE SWITCHED
1100	AI-16	Q			DTE
1101	AI-16	Q			GROUP
1110	AI-16	C			NETWORK
1120	AI-16	C	Q		LINE
1130	CM-1/2	C	Q		CHANNELS
	DU-2				range beginning
	DU-2				range end (none if same as beginning)
1131	DU-2	C	Q	RO	MAXIMUM CHANNELS
1132	DU-2	C	Q		MAXIMUM CIRCUITS
1140	DU-2	C			DEFAULT DATA
1141	DU-1	C			DEFAULT WINDOW
1150	DU-2	C			MAXIMUM DATA
1151	DU-1	C			MAXIMUM WINDOW
1152	DU-1	C			MAXIMUM CLEARS
1153	DU-1	C			MAXIMUM RESETS
1154	DU-1	C			MAXIMUM RESTARTS
1160	DU-1	C			CALL TIMER
1161	DU-1	C			CLEAR TIMER
1162	DU-1	C			RESET TIMER
1163	DU-1	C			RESTART TIMER
1170	AI-16	C	Q		DTE (qualified by GROUP)
1171	DU-2	C	Q		NUMBER (qualified by GROUP)
1172	C-1	C	Q		TYPE (qualified by GROUP)

The values for STATE are:

Value	Meaning
0	ON
1	OFF
2	SHUT

The values for Substate are:

Value	Meaning
0	Running
1	Sync
2	Unsync

The value for TYPE is:

Value	Meaning
1	BILATERAL

7.7.7 X.25 Server Module Parameters

The following table specifies the X.25 server module parameters.

Table 17
X.25 Server Module Parameters

Type Number	Data Type	Inf. Type	App. Rest.	Set. Rest.	NCP Keywords
100	DU-2	C			COUNTER TIMER
200	DU-2	S*		RO	ACTIVE CIRCUITS
300	AI-16	Q			DESTINATION
310	DU-2	C			MAXIMUM CIRCUITS
320	CM-1/2	C	Q		NODE
	DU-2				node address
	AI-6				node name (optional)
330	AI-39	C	Q		USER user
331	AI-39	C	Q	WO	PASSWORD (to set)
331	C-1	C	Q	RO	PASSWORD (to read)
332	AI-39	C	Q		ACCOUNT
340	CM-1/2	C	Q		OBJECT
	DU-1				object number
	AI-16				object name
350	DU-1	C	Q		PRIORITY
351	HI-16	C	Q		CALL MASK
352	HI-16	C	Q		CALL VALUE
353	AI-16	C	Q		GROUP
354	AI-16	C	Q		NUMBER
355	CM-1/2	C	Q		SUBADDRESSES
	DU-2				range beginning
	DU-2				range end (none if same as beginning)

An output, the PASSWORD parameter is not present if there is no password. If it is present, its value is:

Value	Meaning
0	Password set

7.8 Module Counters

7.8.1 X.25 Protocol Module Counters

The following table specifies the X.25 protocol module local DTE counters.

Table 18
X.25 Protocol Module Counters

Type Number	Bit Width	Standard Text
0	16	Seconds since last zeroed
1000	32	Bytes received
1001	32	Bytes sent
1010	32	Data blocks received
1011	32	Data blocks sent
1200	16	Calls received
1201	16	Calls sent
1210	16	Fast selects received
1211	16	Fast selects sent
1220	16	Maximum switched circuits active
1221	16	Maximum channels active
1230	16	Received call resource errors
1240	8	Locally initiated resets
1241	8	Remotely initiated resets
1242	8	Network initiated resets
1250	8	Restarts

7.8.2 X.25 Server Module Counters

The following table specifies the X.25 server module counters.

Table 19
X.25 Server Module Counters

Type Number	Bit Width	Standard Text
0	16	Seconds since last zeroed
200	16	Maximum circuits active
210	8	Incoming calls rejected, no resources
211	8	Logical links rejected, no resources

7.9 Node Parameters

The following table specifies the node parameters:

Table 20
Node Parameters

Param. Type Number	NICE Data Type	Inf. Type	Appl. Rest.	Set. Rest.	NCP Keywords
0	C-1	S*	E,R		STATE
10	HI-6	S	E	RO	PHYSICAL ADDRESS
100	AI-32	C*	E		IDENTIFICATION
101	CM-3	C	E	RO	MANAGEMENT VERSION
	DU-1				version number
	DU-1				ECO number
	DU-1				User ECO number
110	AI-16	C	A		SERVICE CIRCUIT
111	H-8	C	A		SERVICE PASSWORD
112	C-1	C	A		SERVICE DEVICE
113	C-1	C	A		CPU
114	HI-6	C	A		HARDWARE ADDRESS
115	C-1	C	A		SERVICE NODE VERSION
120	AI-255	C	A		LOAD FILE
121	AI-255	C	A		SECONDARY LOADER
122	AI-255	C	A		TERTIARY LOADER
123	AI-255	C	A		DIAGNOSTIC FILE
125	C-1	C	A		SOFTWARE TYPE
126	AI-16	C	A		SOFTWARE IDENTIFICATION
130	AI-255	C	A		DUMP FILE
131	AI-255	C	A		SECONDARY DUMPER
135	O-4	C	A		DUMP ADDRESS
136	DU-4	C	A		DUMP COUNT
140	CM-1/2	C	A,E	RO	HOST
	DU-2				Node address
	AI-6				Node name (optional)
141	(node-id)		A,E	WO	HOST
150	DU-2	C	E	LO	LOOP COUNT
151	DU-2	C	E	LO	LOOP LENGTH
152	C-1	C	E	LO	LOOP WITH
153	HI-6		E	LO	LOOP ASSISTANT PHYSICAL ADDRESS
154	C-1	C	E	LO	LOOP HELP
155	(node-id)		E	LO	LOOP NODE
156	(node-id)		E	LO	LOOP ASSISTANT NODE
160	DU-2	C	E,R		COUNTER TIMER
500	(id-string)		E,R	WO	NAME
501	AI-16	C*	Q,L,N		CIRCUIT
502	DU-2	n/a	E	WO	ADDRESS
510	DU-2	C	E		INCOMING TIMER
511	DU-2	C	E		OUTGOING TIMER
600	DU-2	S*	E,R	RO	ACTIVE LINKS
601	DU-2	S*	R	RO	DELAY
700	CM-3	C	E	RO	NSP VERSION
	DU-1				version number
	DU-1				ECO number
	DU-1				User ECO number
710	DU-2	C	E		MAXIMUM LINKS
720	DU-1	C	E		DELAY FACTOR

721	DU-1	C	E		DELAY WEIGHT
722	DU-2	C	E		INACTIVITY TIMER
723	DU-2	C	E		RETRANSMIT FACTOR
810	C-1	S	A	RO	TYPE
820	DU-2	S	R	RO	COST
821	DU-1	S	R	RO	HOPS
822	AI-16	S*	R	RO	CIRCUIT
830	CM-1/2	S*	R	RO	NEXT NODE
	DU-2				Node address
	AI-6				Node name (optional)
900	CM-3	C	E	RO	ROUTING VERSION
	DU-1				Version number
	DU-1				ECO number
	DU-1				User ECO number
901	C-1	C	E		TYPE
910	DU-2	C	E		ROUTING TIMER
911	CM-1/2	C	E		SUBADDRESSES
	DU-2				range beginning
	DU-2				range end (none if same as beginning)
912	DU-2	C	E		BROADCAST ROUTING TIMER
920	DU-2	C	E		MAXIMUM ADDRESS
921	DU-2	C	E		MAXIMUM CIRCUITS
922	DU-2	C	E		MAXIMUM COST
923	DU-1	C	E		MAXIMUM HOPS
924	DU-1	C	E		MAXIMUM VISITS
925	DU-1	C	E		MAXIMUM AREA
926	DU-2	C	E		MAXIMUM BROADCAST NONROUTERS
927	DU-2	C	E		MAXIMUM BROADCAST ROUTERS
928	DU-2	C	E		AREA MAXIMUM COST
929	DU-1	C	E		AREA MAXIMUM HOPS
930	DU-2	C	E		MAXIMUM BUFFERS
931	DU-2	C	E		BUFFER SIZE
932	DU-2	C	E		SEGMENT BUFFER SIZE

The values for STATE are:

Value	Keyword	Node
0	ON	Executor
1	OFF	Executor
2	SHUT	Executor
3	RESTRICTED	Executor
4	REACHABLE	Destination
5	UNREACHABLE	Destination

The values for SERVICE DEVICE are the same as found in the DNA Low Level Maintenance Operation specification. They are also defined in section 7.4, following Table 10, under Communication DEVICE mnemonics.

The values for CPU are:

Value	Type
0	PDP8
1	PDP11
2	DECSYSTEM1020
3	VAX

The values for SOFTWARE TYPE are:

Value	Program Type
0	SECONDARY LOADER
1	TERTIARY LOADER
2	SYSTEM

The values for HOST are:

NODE ADDRESS (2): B

Host node address.

NODE NAME (I-6): A

Host node name, zero length if none.

The values for LOOP WITH are:

Type	Contents
0	ZEROES
1	ONES
2	MIXED

The values for LOOP HELP are:

Type	Contents
0	TRANSMIT
1	RECEIVE
2	FULL

The values for executor and adjacent node TYPE are:

Value	Keyword
0	ROUTING III
1	NONROUTING III
3	AREA
4	ROUTING IV
5	NONROUTING IV

The values for SERVICE NODE VERSION are:

Value	Keyword
0	PHASE III
1	PHASE IV

7.10 Node Counters

Table 21, below, lists the node counters. The definition of each counter and the way it is to be incremented is given in the functional specifications for the layer containing the counter.

Node counters are specified for the following layers only:

Layer	Type Number Range
Network Management	000 - 099
End Communication	600 - 700
Routing	900 - 999

Table 21
Node Counters

Appl.	Type Number	Bit width	Standard Text
DN	0	16	Seconds Since Last Zeroed
DN	600	32	User Bytes Received
DN	601	32	User Bytes Sent
DN	602	32	User Messages Received
DN	603	32	User Messages Sent
DN	608	32	Total Bytes Received
DN	609	32	Total Bytes Sent
DN	610	32	Total Messages Received
DN	611	32	Total Messages Sent
DN	620	16	Connects Received
DN	621	16	Connects Sent
DN	630	16	Response Timeouts
DN	640	16	Received Connect Resource Errors
E	700	16	Maximum Logical Links Active

E	900	8	Aged Packet Loss
E	901	16	Node Unreachable Packet Loss
E	902	8	Node Out-of-Range Packet Loss
E	903	8	Oversized Packet Loss
E	910	8	Packet Format Error
E	920	8	Partial Routing Update Loss
E	930	8	Verification Reject

7.11 Area Parameters

The following table specifies the area parameters:

Table 21a
Area Parameters

Param. Type Number	NICE Data Type	Appl. Type	Set. Rest.	NCP Keywords
0	C-1	S*	RO	STATE
820	DU-2	S	RO	COST
821	DU-1	S	RO	HOPS
822	AI-16	S*	RO	CIRCUIT
830	CM-1/2	S*	RO	NEXT NODE
	DU-2			Node address
	AI-6			Node name (optional)

The values for STATE are:

Value	Keyword	Node
4	REACHABLE	Destination
5	UNREACHABLE	Destination

7.12 Event Definitions

Table 22, following, defines the event classes. The event class as shown in Table 22 is a composite of the system type and the system specific event class.

Table 22
Event Classes

Event Class	Description
0	Network Management Layer
1	Applications Layer

2	Session Control Layer
3	End Communication Layer
4	Routing Layer
5	Data Link Layer
6	Physical Link Layer
7-31	Reserved for other common classes
32-63	RSTS System specific
64-95	RSX System specific
96-127	TOPS 10/20 System specific
128-159	VMS System specific
160-191	RT System specific
192-223	CT System specific
224-255	Communication Server specific
256-479	Reserved for future use
480-511	Customer specific

In the following descriptions, an entity related to an event indicates that the event can be filtered specific to that entity. Binary logging data is formatted under the same rules as the data in NICE data (see Section 7.1).

Table 23 shows the events for each class.

Table 23
Events

Class	Type	Entity	Standard Text	Event Parameters and Counters
0	0	none	Event records lost	none
0	1	node	Automatic node counters	Node counters
0	2	line	Automatic line counters	Line counters
0	3	circuit	Automatic service	Service Status Node Filespec Software type
0	4	line	Line counters zeroed	Circuit counters
0	5	node	Node counters zeroed	Node counters
0	6	circuit	Passive loopback	Operation
0	7	circuit	Aborted service request	Reason Node
0	8	any	Automatic counters	Qualifier Counters DTE
0	9	any	Counters zeroed	Qualifier Counters DTE
2	0	none	Local node state change	Reason Old state New state

2	1	none	Access control reject	Source node Source process Destination process User Password Account
3	0	none	Invalid message	Message Source Node
3	1	none	Invalid flow control	Message Source Node Current flow control
3	2	node	Data base reused	NSP node counters
4	0	none	Aged packet loss	Packet header
4	1	circuit	Node unreachable packet loss	Packet header Adjacent node
4	2	circuit	Node out-of-range packet loss	Packet header Adjacent node
4	3	circuit	Oversized packet loss	Packet header Adjacent node
4	4	circuit	Packet format error	Packet beginning Adjacent node
4	5	circuit	Partial routing update loss	Packet header Highest address Adjacent node
4	6	circuit	Verification reject	Node
4	7	circuit	Circuit down, circuit fault	Reason Adjacent node
4	8	circuit	Circuit down	Reason Packet header Adjacent node
4	9	circuit	Circuit down, operator initiated	Reason Packet header Adjacent node
4	10	circuit	Circuit up	Adjacent node
4	11	circuit	Initialization failure, line fault	Reason
4	12	circuit	Initialization failure, software fault	Reason Packet header
4	13	circuit	Initialization failure, operator fault	Reason Packet header Received version
4	14	node	Node reachability change	Status
4	15	circuit	Adjacency up	Adjacent node
4	16	circuit	Adjacency rejected	Adjacent node Reason
4	17	area	Area reachability change	Status
4	18	circuit	Adjacency down	Reason Packet header Adjacent node
4	19	circuit	Adjacency down, operator initiated	Reason Packet header Adjacent node
5	0	circuit	Locally initiated state change	Old state New state

5	1	circuit	Remotely initiated state change	Old state New state
5	2	circuit	Protocol restart received in maintenance mode	none
5	3	circuit	Send error threshold	Circuit counters
5	4	circuit	Receive error threshold	Circuit counters
5	5	circuit	Select error threshold	Circuit counters
5	6	circuit	Block header format error	Header (optional)
5	7	circuit	Selection address error	Selected tributary Received tributary Previous tributary
5	8	circuit	Streaming tributary	Tributary status Received tributary
5	9	circuit	Local buffer too small	Block length Buffer length
5	10	module	Restart (X.25 protocol)	DTE Cause Diagnostic
5	11	module	State change (X.25 protocol)	DTE Reason Old state New state
5	12	module	Retransmit maximum exceeded	DTE Parameter type
5	13	line	Initialization failure	none
5	14	line	Send failed	Failure reason Distance
5	15	line or circuit	Receive failed	Failure reason Ethernet header
5	16	line	Collision detect check failed	
5	17	module	DTE up	DTE
5	18	module	DTE down	DTE
6	0	line	Data set ready transition	New state
6	1	line	Ring indicator transition	New state
6	2	line	Unexpected carrier transition	New state
6	3	line	Memory access error	Device register
6	4	line	Communications interface error	Device register
6	5	line	Performance error	Device register

7.13 Event Parameters

The following parameter types are defined for the Network Management Layer (class 0):

Table 24
Network Management Layer Event Parameters

Type	Data Type	Keywords
0	C-1	SERVICE
1	CM-1/2/3	STATUS (as in NICE)
	C-1	Return code
	C-2	Error detail (optional if no error message
	AI-72	Error message (optional)
2	C-1	OPERATION
3	C-1	REASON
4	CM-2	Qualifier
	C-2	Parameter type
	AI-16	ID string
5	CM-1/2	NODE
	DU-2	Node address
	AI-6	Node name (optional)
6	AI-16	DTE
7	AI-255	Filespec
8	C-1	SOFTWARE TYPE

The values for SERVICE are:

value	Keyword
0	LOAD
1	DUMP

The values for Return code added interpretation are:

Value	Keyword
0	REQUESTED
>0	SUCCESSFUL
<0	FAILED

The values for OPERATION are:

Value	Keyword
0	INITIATED
1	TERMINATED

The values for REASON are:

Value	Reason
0	Receive timeout
1	Receive error
2	Line state change by higher level
3	Unrecognized request
4	Line open error

The values for SOFTWARE TYPE are the same as those in Node Parameters.

The following parameter types are defined for the Session Control layer (class 2):

Table 25
Session Control Layer Event Parameters

Type	Data Type	Keywords
0	C-1	REASON
1	C-1	OLD STATE
2	C-1	NEW STATE
3	CM-1/2	SOURCE NODE
	DU-2	node address
	AI-6	node name (optional if none)
4	CM-1/2/3/4	SOURCE PROCESS
	DU-1	Object type
	DU-2	Group code (if specified and process name present)
	DU-2	User code (if specified and group code present)
	AI-16	Process name (if specified)
5	CM-1/2/3/4	DESTINATION PROCESS
		Same as for SOURCE PROCESS
6	AI-39	USER
7	C-1	PASSWORD
8	AI-39	ACCOUNT

The values for REASON are:

Value	Meaning
0	Operator command
1	Normal operation

The values for OLD STATE and NEW STATE are:

Value	Meaning
0	ON
1	OFF
2	SHUT
3	RESTRICTED

A value of zero for PASSWORD indicates a password was set. Absence of the parameter indicates no password was set.

The following parameter types are defined for the End Communication layer (class 3):

Table 26
End Communication Layer Event Parameters

Type	Data Type	Keywords
0	CM-4	MESSAGE
	H-1	Message flags
	DU-2	Destination logical link address
	DU-2	Source logical link address
	HI-6	Message type dependent data
1	DS-1	CURRENT FLOW CONTROL REQUEST COUNT
2	CM-1/2	SOURCE NODE
	DU-2	Node address
	AI-6	Node name (optional)

The following parameter types are defined for the Routing layer (class 4):

Table 27
Routing Layer Event Parameters

Type	Data Type	Keywords
0	CM-2/4	PACKET HEADER (non-Ethernet)
	H-1	Message flags
	DU-2	Destination node address (not present for control packet)
	DU-2	Source node address
	DU-1	Visit count (not present for control packet)

0	CM-11	PACKET HEADER (Ethernet)
	H-1	Message flags
	DU-1	Destination area
	DU-1	Destination subarea
	HI-6	Destination Ethernet address
	DU-1	Source area
	DU-1	Source subarea
	HI-6	Source Ethernet address
	DU-1	Next area router
	DU-1	Visit count
	H-1	Service class
	DU-1	Protocol type
1	HI-6	PACKET BEGINNING
2	DU-2	HIGHEST ADDRESS
3	CM-1/2	NODE
	DU-2	node address
	AI-6	node name (optional if none)
4	CM-1/2	EXPECTED NODE
	DU-2	node address
	AI-6	node name (optional if none)
5	C-1	REASON
6	CM-3	RECEIVED VERSION
	DU-1	Version number
	DU-1	ECO number
	DU-1	User ECO number
7	C-1	STATUS
8	CM-1/2	ADJACENT NODE
	DU-2	node address
	AI-6	node name (optional if none)

The values for REASON are:

Value	Meaning
0	Circuit synchronization lost
1	Data errors
2	Unexpected packet type
3	Routing update checksum error
4	Adjacency address change
5	Verification receive timeout
6	Version skew
7	Adjacency address out of range
8	Adjacency block size too small
9	Invalid verification seed value
10	Adjacency listener receive timeout
11	Adjacency listener received invalid data
12	Call failed
13	Verification password required from Phase III node
14	Dropped by adjacent node

The values for STATUS are:

Value	Meaning
0	REACHABLE
1	UNREACHABLE

The following parameter types are defined for the Data Link layer (class 5):

Table 28
Data Link Layer Event Parameters

Type	Data Type	Keywords
0	C-1	OLD STATE
1	C-1	NEW STATE
2	HI-6	HEADER
3	DU-1	SELECTED TRIBUTARY
4	DU-1	PREVIOUS TRIBUTARY
5	C-1	TRIBUTARY STATUS
6	DU-1	RECEIVED TRIBUTARY
7	DU-2	BLOCK LENGTH
8	DU-2	BUFFER LENGTH
9	AI-16	DTE
10	C-1	REASON
11	C-1	OLD STATE (for event 5.12)
12	C-1	NEW STATE (for event 5.12)
13	C-2	PARAMETER TYPE
14	DU-1	CAUSE
15	DU-1	DIAGNOSTIC
16	C-1	FAILURE REASON
17	DU-2	DISTANCE
18	CM-3	ETHERNET HEADER
	HI-6	Destination address
	HI-6	Source address
	HI-2	Protocol type
19	NC	HARDWARE STATUS

The values for OLD STATE and NEW STATE are:

Value	Meaning
0	HALTED
1	ISTRT
2	ASTRT
3	RUNNING
4	MAINTENANCE

The values for FAILURE REASON are:

Value	Meaning
0	Excessive collisions
1	Carrier check failed
2	(OBSOLETE)
3	Short circuit
4	Open circuit
5	Frame too long
6	Remote failure to defer
7	Block check error
8	Framing error
9	Data overrun
10	System buffer unavailable
11	User buffer unavailable
12	Unrecognized frame destination

The values for REASON are:

Value	Meaning
0	Operator command
1	Normal operation

The values for OLD STATE and NEW STATE are:

Value	Meaning
0	ON
1	OFF
2	SHUT

The values for TRIBUTARY STATUS are:

Value	Meaning
0	Streaming
1	Continued send after timeout
2	Continued send after deselect
3	Ended streaming

The following parameter types are defined for the Physical Link layer (class 6):

Table 29
Physical Link Layer Event Parameters

Type	Data Type	Keywords
0	NC	DEVICE REGISTER
1	C-1	NEW STATE

The values for NEW STATE are:

Value	Meaning
0	OFF
1	ON

APPENDIX A

VERSION COMPATIBILITY

This appendix describes the mapping necessary to maintain compatibility between different versions of Network Management. It has separate sections for 2.0 and 3.0 and for 3.0 and 4.0. The mapping between 2.0 and 4.0 is simply the combination of the two sections.

A.1 Versions 2.0 and 3.0

There are two cases where compatibility is at issue: version 3.0 NCP with version 2.0 Listener and version 2.0 NCP with version 3.0 Listener. In both cases, it is the responsibility of the module with the later version to provide compatibility, if such compatibility is part of the individual products requirements. When possible, functions are to be mapped between versions, but new version 3.0 functions are not available from version 2.0 modules.

Version 3.0 NCP supports only the version 3.0 command syntax. If it is to be compatible with a version 2.0 Listener, it must map 3.0 commands to 2.0 NICE protocol and 2.0 responses to 3.0 output. Any 2.0 responses that do not map to 3.0 output should be handled through the normal unrecognized response logic.

Version 3.0 Listener supports version 2.0 commands by mapping them to 3.0 functions. It then maps the 3.0 responses into 2.0 responses. In cases where a response cannot be mapped and does not conflict with a 2.0 response, it can be returned unmapped to be handled by the 2.0 systems normal unrecognized response logic.

In the following mapping specifications, parameters that did not change between version 2.0 and 3.0 are not mentioned. All mappings apply to both commands and responses unless otherwise stated.

A.1.1 Module Entity

None of the 3.0 module entity functions can be mapped to version 2.0.

A.1.2 Node Entity

The following mappings apply to the node entity.

Version 3.0 Parameter	Version 2.0 Parameter
CIRCUIT	LINE
SERVICE CIRCUIT	SERVICE LINE
MAXIMUM CIRCUITS	MAXIMUM LINES

A.1.3 Logging Entity

For the logging entity, the only mapping is for the entity identification that goes with specific filters.

For a version 3.0 NCP with a version 2.0 Listener, version 3.0 circuit and line identifications become version 2.0 line identifications.

For a version 2.0 NCP with a version 3.0 Listener, the version 2.0 line identification becomes a version 3.0 circuit identification and, where necessary, is applied to the associated version 3.0 line by implication.

A.1.4 Circuit and Line Entities

The following statements apply to the case of a version 3.0 NCP with a version 2.0 Listener.

The following version 3.0 circuit parameters map to the version 2.0 line parameters of the same name.

STATE
Substate
SERVICE
COUNTER TIMER

LOOPBACK NAME
ADJACENT NODE
BLOCK SIZE
COST
TYPE TRIBUTARY

None of the other version 3.0 circuit parameters can be mapped to the version 2.0 line.

The following version 3.0 line parameters map to the version 2.0 line parameters of the same name.

DUPLEX
CONTROLLER
SERVICE TIMER

The following version 3.0 line parameters map to version 2.0 line parameters of a different name.

Version 3.0	Version 2.0
PROTOCOL	TYPE
RETRANSMIT TIMER	NORMAL TIMER

None of the other version 3.0 line parameters map to version 2.0 line parameters.

The version 3.0 LOOP CIRCUIT and LOOP LINE commands both map to version 2.0 line as do the SHOW, LIST, TRIGGER, LOAD, and DUMP commands.

The following statements apply to the case of a version 2.0 NCP with a version 3.0 Listener.

Version 2.0 commands can only be applied to version 3.0 circuits. Their application to version 3.0 lines occurs only by implication. The version 3.0 line parameters are referenced indirectly by the line's association with the circuit and thus appear to the version 2.0 system to belong to the circuit. Version 3.0 lines that are not directly associated with a circuit are not visible to the version 2.0 system.

The following version 2.0 line parameters map to the version 3.0 circuit parameters of the same name.

STATE
Substate
SERVICE
COUNTER TIMER
LOOPBACK NAME
ADJACENT NODE
BLOCK SIZE
COST
TRIBUTARY

The following version 2.0 line parameters map to the version 3.0 line parameters of the same name.

CONTROLLER
DUPLEX
SERVICE TIMER

The following version 2.0 line parameters map to version 3.0 line parameters of a different name.

Version 2.0	Version 3.0
TYPE	PROTOCOL
NORMAL TIMER	RETRANSMIT TIMER

A.1.5 Event Logging

A version 3.0 event transmitter does not send new version 3.0 events to a version 2.0 event receiver. Version 3.0 circuit events become version 2.0 line events unless they are X.25 specific, in which case they are not sent.

A version 3.0 event receiver translates version 2.0 line events for data link and above into circuit events.

A.2 Versions 3.0 and 4.0

The only mapping between version 3.0 and version 4.0 is in the handling of area numbers. Version 3.0 does not recognize the high byte of the address field as the area number. Therefore, a version 4.0 node sending NICE messages to a version 3.0 node must zero the area portion of any address field for nodes in the same area as the version 3.0 node.

Otherwise, the normal responses to unrecognized functions, options, parameters, etc. will suffice.

APPENDIX B

MINIMUM SUBSET

The intent of the Network Management minimum subset is to ensure that all DNA products will provide sufficient Network Management capabilities to manage both individual nodes and the network as a whole. It places strict requirements on the implementers of the Network Management architecture and provides guidelines for configuring nodes and networks.

The minimum subset must be interpreted in the light of specific product requirements. The Network Management requirements of an unattended routing node in a centrally managed network are obviously different from those of an attended non-routing node in a network with fully distributed management. Before the proper minimum subset can be determined for a particular network product, its potential uses as a network node must be defined.

The minimum subset defines those capabilities that must be provided by the product implementers. If the product is capable of widely different responsibilities within the network, the implementers should allow for different Network Management subsets within the product, so that the configurations that require fewer functions are not burdened by the requirements of more complex situations.

The minimum subset contains the smallest set of Network Management functions necessary to diagnose network problems that have a general level of negative effect, and remove the communications paths or nodes that are at fault. It does not contain sufficient functions to provide complete fault diagnosis, network planning, or general control.

Following are the minimum, general capabilities that are required for node and network management. Unattended nodes, nodes with no console, or nodes that are to be remotely controlled must provide these capabilities via the NICE protocol. Nodes that are to be locally managed must provide the capabilities via NCP. Nodes that are to be both remotely and locally controlled must provide these capabilities via both NCP and the NICE protocol.

Each capability below is followed by the NICE messages and the NCP syntax that implements it.

1. Display and zero of all existing counters.

   The Data Link layer counters must exist on all multipoint control nodes and on at least one end of each point-to-point link.

   All DECnet nodes must support remote and local read capabilities of the Ethernet data link counters via the NICE protocol.

   NICE messages:

   Read information message, circuit counter and line counter options.

   Zero counters message, circuit and line options.

   NCP commands:

   SHOW CIRCUIT circuit-id COUNTERS

   SHOW LINE line-id COUNTERS

   ZERO CIRCUIT circuit-id COUNTERS

   ZERO LINE line-id COUNTERS

   The Routing layer counters must exist at all routing nodes.

   NICE messages:

   Read information message, circuit counters and node counters options.

   Zero counters message, circuit counters and node counters options.

   NCP commands:

   SHOW CIRCUIT circuit-id COUNTERS

   SHOW NODE node-id COUNTERS

   ZERO CIRCUIT circuit-id COUNTERS

   ZERO NODE node-id COUNTERS

   The End Communication layer counters must exist on one node of any pair of nodes that will communicate via logical links and that are more than one hop away from each other. Communicating nodes that are only one hop away from each other are not required to support the End Communication Layer counters.

   NICE messages:

   Read information message, node counters option.

Zero counters message, node counters option.

NCP commands:

```
SHOW NODE node-id COUNTERS

ZERO NODE node-id COUNTERS
```

All other counters not mentioned above must exist if the node provides the service for which the counters are defined. For example a node that provides an X.25 gateway server must provide the corresponding counters.

2. Display and control of Routing events for all routing nodes.

   Display, control and logging of routing event messages are optional on non-routing nodes. Routing nodes must be capable of sending routing event messages to at least a sink node, which can be set to be any node in the network. (Note: The routing node sending the message is not required to serve as a sink node.) Any network incorporating routing nodes must have at least one node that may serve as a sink for events.

   NICE messages:

   Change parameter message, set and clear all logging parameters for FILE, CONSOLE or MONITOR.

   Read information message, logging summary option with sink node qualifier.

   NCP commands:

```
SET LOGGING sink-type   EVENTS event-list -
                               [source-qualifier] -
                               [sink-node]
                        NAME sink-name
                        STATE sink-state

CLEAR LOGGING sink-type EVENTS event-list -
                               [source-qualifier] -
                               [sink-node]
                        NAME

SHOW LOGGING sink-type  SUMMARY [sink-node]
```

3. Test communication using the node level logical link loop test.

   Note that this implies that all nodes must implement the Loopback Mirror.

   NICE messages:

   Test message, node level loop test option.

Change parameter message, set and clear node option, circuit and name parameters.

Read information message, node summary option.

NCP commands:

```
SET NODE node-id           NAME node-name
                           CIRCUIT circuit-id

CLEAR NODE node-id         NAME
                           CIRCUIT

SHOW NODE node-id          SUMMARY

LOOP NODE node-id [access-control] [WITH block-type]
                                   [COUNT count]
                                   [LENGTH length]
```

4. Disable a point-to-point communications link by setting its state to OFF.

NICE messages:

Change parameter message, set circuit option, state parameter, off value.

NCP commands:

```
SET CIRCUIT circuit-id STATE OFF
```

5. Disable a node by setting its state to OFF.

NICE messages:

Change parameter message, set node option, state parameter, off value.

NCP commands:

```
SET NODE node-id STATE OFF
```

6. Display minimal information about a point-to-point communications link or node.

NICE messages:

Read information message, circuit and node summary options.

NCP commands:

```
SHOW CIRCUIT circuit-id SUMMARY
```

SHOW NODE node-id SUMMARY

7. From NCP, send command to remotely managed nodes from central management nodes. Note that any product that could be a central management node must be able to parse all possible NCP commands and format all possible responses. This must be true even though the product does not itself implement the particular options.

   NCP commands:

      TELL node-id [access-control] command

      SET EXECUTOR NODE node-id [access-control]

      CLEAR EXECUTOR NODE

8. Display and control minimal multipoint control parameters for lines running protocol type DDCMP control.

   NICE messages:

      Change parameter message, set line option, dead timer and delay timer parameters.

      Read information message, line summary option.

   NCP commands:

```
SET LINE line-id DEAD TIMER milliseconds
                 DELAY TIMER milliseconds
```

      SHOW LINE line-id SUMMARY

It is the responsibility of Product Management to set the product requirements so that the applicability of the minimum subset can be determined.

It is the responsibility of Development to build products that can be configured to minimum subset requirements.

It is the responsibility of Software Services or the user to configure nodes so that the requirements are met for the network. This has to be done while taking into account the specific characteristics of the node and network involved.

APPENDIX C

STATE MAPPING TABLES

The following tables relate the Network Management controllable/observable states to the Network Management link states and to the states from the actual specifications for the high level users and data link protocols.

The high level circuit users are Routing and the X.25 Gateway Server. The high level line users are point-to-point and multi-point DDCMP and the X.25 protocol handler.

Mapping Number	Net. Man. State	Net. Man. Substate	D.L. Serv. State	High Lev. State	D. L. State
1	CLEARED	N/A	off	off	off
2	OFF	N/A	off	off	off
3	ON	running	passive	run	run
4	ON	STARTING	passive	start	run
5	ON	synchronizing	passive	start	synch
6	ON	FAILED	passive	fail	-
7	ON	REFLECTING	refl	start	maint
8	ON	LOADING	pass-open	off	maint
9	ON	DUMPING	pass-open	off	maint
10	ON	LOOPING	pass-open	off	maint
11	ON	TRIGGERING	pass-open	off	maint
12	ON	AUTOSERVICE	closed	off	maint
13	ON	REFLECTING	clos-refl	off	maint

14	ON	AUTOLOADING	open	off	maint
15	ON	AUTODUMPING	open	off	maint
16	ON	AUTOTRIGGERING	open	off	maint
17	SERVICE	idle	closed	off	maint
18	SERVICE	REFLECTING	clos-refl	off	maint
19	SERVICE	LOADING	open	off	maint
20	SERVICE	DUMPING	open	off	maint
21	SERVICE	TRIGGERING	open	off	maint
22	SERVICE	LOOPING	open	off	maint

The following tables relate the internal states from each component's own specification to the Network Management states. This relationship is indicated through the mapping numbers which correspond to the preceding table.

The following mappings apply when the circuit user or owner is EXECUTOR.

Mapping Number	Routing State	DDCMP State	Perm Cir. X.25 P.L. State	Inc. Cir. X.25 P.L. State	Out. Cir. X.25 P.L. State
1-2	OP HA	halt	unaloc unun	listening clearing cleared	open clearing cleared
3	RU	run	running	running	running
4	LR TI TV TC	run	running	running	running
5	DS	halt astrt istrt	synch unsynch no-com	synch unsynch no-com listening called taken	synch unsynch no-com calling
6	HA	N/A	N/A	listening	open
7-22	OF	maint	N/A	N/A	N/A

	HA	
8-9	N/A	
10	halt	maint
11-12	N/A	
13	halt	maint
14-16	N/A	
17-18	halt	maint
19-21	N/A	
22	halt	maint

The following table applies when the line protocol is LAPB and the line belongs to the X.25 protocol handler module.

Mapping Number	Low Level Link State	Low Level Port State
1-2	halted disconnecting	unsync unun
3	inf. trans.	running
4	inf. trans.	running
5	connecting re-synch	unsynch synch
6-9	N/A	
10	loopback	unsynch
11-16	N/A	
17	loopback	unload
18-21	N/A	
22	loopback	running

APPENDIX D

X.25 NATIVE ONLY SUBSET

In the case of a system that only provides native mode X.25 usage, many Network Management functions do not apply. These are the functions that relate to a node that is part of a DECnet network. The functions that do apply should still be available through NCP but are not subject to any architectural minimum subset.

The following functions apply to an X.25 native only system.

1. For the circuit entity, the following parameters:

   CHANNEL
   CONNECTED OBJECT
   COUNTER TIMER
   DTE
   MAXIMUM BLOCK
   MAXIMUM WINDOW
   STATE
   Substate
   USER

2. The X.25 permanent virtual circuit counters.

3. For the line entity, the following parameters.

   CLOCK
   CONTROLLER
   COUNTER TIMER
   DEVICE
   HOLDBACK TIMER
   MAXIMUM DATA
   MAXIMUM RETRANSMITS
   MAXIMUM WINDOW
   RETRANSMIT TIMER
   SERVICE
   STATE
   Substate

4. The LAPB line counters.

5. The following Data Link events.

   Locally initiated state change
   Remotely initiated state change

6. For the logging entity, all parameters except SINK NODE.

7. For module X25-ACCESS, the NETWORK parameter.

8. For module X25-PROTOCOL, all parameters, counters, and events.

9. For module X25-SERVER, all counters, and all parameters except the following.

   ACCOUNT
   NODE
   PASSWORD
   USER

10. The following NCP commands, supporting the above mentioned entities and parameters.

    CLEAR
    DEFINE
    PURGE
    LIST
    LOOP LINE
    SET
    SHOW
    ZERO

APPENDIX E

MEMORY IMAGE FORMATS AND FILE CONTENTS

Since the PDP-8, PDP-11, VAX-11, and DECsystem-10, or DECSYSTEM-20 memory addressing requirements differ, different formats are required for memory image data. In each case, it is essential to know the number of bytes that represent the smallest individually addressable memory location. A format summary is provided below.

PDP-8	Each three bytes represents two 12-bit words; that is, the memory address is incremented by two for each three bytes. Byte 1 is the low 8-bits of memory word 1. Byte 2 is the low 8-bits of memory words 1 and 2.
PDP-11 VAX-11	Each byte represents one memory byte. That is, the memory address is incremented with each byte.
DECsystem-10 DECSYSTEM-20	Each five bytes represents one 36-bit word. That is, the memory address is incremented by one for each five bytes. Byte 1 is the highest 8-bits of the word. Bytes 2 through 4 follow. The high 4-bits of byte 5 are discarded.

The files containing memory images for a down-line load or an up-line dump have the same contents. The format may vary from one operating system to another, but the contents are functionally the same in all cases. The minimum control information required is as follows:

- o The type of the target system (PDP-8, PDP-11, VAX-11, DECsystem-10, or DECSYSTEM-20). This is necessary to know how to interpret and update memory address information.

- o Transfer address. This is the startup address for the program. This field is generally meaningless for a dump file.

The image information required is as follows:

- o Memory address. This is the address where image goes for a load or comes from a dump.

- o Block length. Number of memory units in image block.

- Memory image. This is the contiguous block of memory associated with the above address. The format requirements are as specified in Appendix B. The memory image can be of any length.

APPENDIX F

NICE RETURN CODES WITH EXPLANATIONS

This appendix specifies the NICE return codes.

In all cases, the number specified is for the first byte of the return code.

The error detail that sometimes follows the return codes is two bytes long. Since some systems may have trouble implementing the error details, a value of 65,535 (all 16 bits set) in the error detail field means no error detail. In other words, in this case, no error detail will be printed.

If a response message is short terminated after any field, the existing fields may still be interpreted according to the standard format.

A printed error message consists of the standard text for the first byte. If the second and third bytes have a defined value, this is followed by a comma, a blank, and the keyword(s) for the values.

Number	Standard test	Meaning
1	(none)	Success.
2	(none)	The request has been accepted, and separated data responses are coming.
3	(none)	Success, partial reply. More parameters for entity in next message. Can only be embedded in a more/done sequence. Each message still contains fields up to and including ENTITY ID.
-1	Unrecognized function or option	Either the function code or option field requested a capability not recognized by the Local Network Management Function. Also, the error code for function codes 2-14 (Phase

		II), and for system-specific commands when the system type matches the receiving system.
-2	Invalid message format	Message too long or too short (i.e., extra data or not enough data), or a field improperly formatted for data expected.
-3	Privilege violation	The requester does not have the privilege required to perform the requested function.
-4	Oversized Management command message	A message size was too long. The NICE message for the command was too long for the Network Management Listener to receive.
-5	Management program error	A software error occurred in the Network Management software. For example, a function that could not fail did fail. Generally indicates a Network Management software bug.
-6	Unrecognized parameter type	A parameter type included in, for example, a change parameter message not recognized by the Network Management Function. The error detail is the low and high bytes of the parameter type number, interpreted according to the entity involved.
-7	Incompatible Management version	The function requested cannot be performed because the Network Management Version skew between the command source and the command destination is too great.
-8	Unrecognized component	An entity (component) was not known to the node. The error detail contains the entity type number.*
-9	Invalid identification	The format of an entity identification was invalid. For example, a node name with no alpha character, or KNOWN used where not allowed. The error detail contains the entity type number.*

-10	Line communication error	Error in transmit or receive on a line. Can only occur during directed use of the Data Link user interface.
-11	Component in wrong state	An entity (component) was in an unacceptable state. For example, a down-line load attempted over a line that is OFF, or a node name to be used for a loop node already assigned to a node address. The error detail contains the entity type number.*
-13	File open error	A file could not be opened. The error detail is defined as follows: Value Keywords 0 PERMANENT DATABASE 1 LOAD FILE 2 DUMP FILE 3 SECONDARY LOADER 4 TERTIARY LOADER 5 SECONDARY DUMPER 6 VOLATILE DATABASE 7 DIAGNOSTIC FILE
-14	Invalid file contents	The data in a file was invalid. The error detail is defined as for error #-13.
-15	Resource error	Some resource was not available. For example, an operating system resource not available.
-16	Invalid parameter value	Improper line-identification type, load address, memory length, etc. The error detail is the low and high bytes of the parameter type number, interpreted according to the entity involved.
-17	Line protocol error	Invalid line protocol message or operation. Can only occur during direct line access. In the case of a line loop test, it indicates that an error was detected during message comparison that should have been caught by the line protocol.

-18	File I/O error	I/O error in a file, such as read error in system image or loader during down-line load. The error detail is defined as for error #-13.
-19	Mirror link disconnected	A successful connect was made to the Loopback Mirror, but the logical link then failed. The error detail is:

Value	Standard text
0	No node name set
1	Invalid node name format
2	Unrecognized node name
3	Node unreachable
4	Network resources
5	Rejected by object
6	Invalid object name format
7	Unrecognized object
8	Access control rejected
9	Object too busy
10	No response from object
11	Remote node shut down
12	Node or object failed
13	Disconnect by object
14	Abort by object
15	Abort by Management
16	Local node shut down

-20	No room for new entry	Insufficient table space for new entry.
-21	Mirror connect failed	A connect to the Network Management Loopback Mirror could not be completed. The error detail is the same as for error #-19.
-22	Parameter not applicable	Parameter not applicable to entity. For example, setting a tributary address for a point-to-point line or an attempt to set a controller to loopback mode when the

		controller does not support that function. The error detail contains the parameter type of the parameter that is not applicable.
-23	Parameter value too long	A parameter value was too long for the implementation to handle. The error detail is the low and high bytes of the parameter type number, interpreted according to the entity involved.
-24	Hardware failure	The hardware associated with the request could not perform the function requested.
-25	Operation failure	A requested operation failed, and there is no more specific error code.
-26	System-specific Management function not supported	Error return for system-specific functions unless the system type is for the system receiving the command. May be further explained by a system-specific error message.
-27	Invalid parameter grouping	The request for changing multiple parameters contained some that cannot be changed with others.
-28	Bad loopback response	A loopback message did not match what was expected, either content or length.
-29	Parameter missing	A required parameter was not included. The error detail is the low and high bytes of the parameter type number, interpreted according to the entity involved.
-128	(none)	No message printed. Done with multiple response commands (e.g., read information for known lines).

*NOTE

Error codes -8, -9, and -11 indicate problems with the primary entity to which a command applies. They may also apply to a secondary entity, such as the line in a LOAD NODE command.

APPENDIX G

NCP COMMAND STATUS AND ERROR MESSAGES

NCP has the following standard status and error messages.

Standard Text	Meaning
	Status Messages
COMPLETE	The command was processed successfully.
FAILED	The command did not execute successfully.
NOT ACCEPTED	The command did not get past syntax and semantic checking. No attempt was made to execute it. The text of the error message may vary as long as the meaning is clearly the same.
	Error Messages
Unrecognized command	The command typed by the user was not recognized.
Unrecognized keyword	Something in the command keyword was not recognized.
Value out of range	A parameter value was out of range. This message may be followed by a comma, a blank and the parameter keyword(s).
Unrecognized value	A parameter value was unrecognizable. This message may be followed by a comma, a blank and the parameter keyword(s).
Not remotely executable	NCP is functionally unable to send a command to a remote node.
Bad management response	The Network Management Access Routines received unrecognizable information.
Listener link disconnected	A successful connect was made to the

	Network Management Listener, but the logical link then failed. Optional error detail is as in NICE error message -19 (Appendix F).
Listener connect failed	A connect to the Network Management Listener could not be completed. The optional error detail is as in NICE error message -21 (Appendix F).
Total parameter data too long	NCP command overflows maximum NICE message for this implementation.
Oversized Management response	NCP could not receive a NICE message because it was too long.

APPENDIX H

JULIAN HALF-DAY ALGORITHMS

The following algorithms will convert to and from a Julian half-day in the range 1 January 1977 through 9 November 2021 as used in the binary format of event logging records.

The algorithms will operate correctly with 16 bit arithmetic. The arithmetic expressions are to be evaluated using FORTRAN operator precedence and integer arithmetic.

In all cases, the input is assumed to be correct, i.e., the day is in the range 1 to maximum for the month, the month is in the range 1-12, the year is in the range 1977-2021 and the Julian half-day is in the range 0-32767.

To convert to Julian half-day:

```
JULIAN = (3055*(MONTH+2)/100-(MONTH+10)/13*2-91
         +(1-(YEAR-YEAR/4*4+3)/4)*(MONTH+10)/13+DAY-1
         +(YEAR-1977)*365+(YEAR-1977)/4)*2
```

To convert from Julian half-day:

```
HALF = JULIAN/2
TEMP1 = HALF/1461
TEMP2 = HALF-TEMP1
YEAR = TEMP2/365
IF TEMP2/1460*1460 = TEMP2 AND (HALF+1)/1460 > TEMP1
  YEAR = YEAR-1
ENDIF
TEMP1 = TEMP2-(YEAR*365)+1
YEAR = YEAR+1977
IF YEAR/4*4 = YEAR
  TEMP2 = 1
ELSE
  TEMP2 = 0
ENDIF
IF TEMP1 > 59+TEMP2
  DAY = DAY+2-TEMP2
ELSE
  DAY = TEMP1
ENDIF
```

```
      MONTH = (DAY+91)*100/3055
      DAY = DAY+91-MONTH*3055/100
      MONTH = MONTH-2
      IF HALF*2 = JULIAN
        HALF = 0
      ELSE
        HALF = 1
      ENDIF
```

The algorithm was certified to work using the following FORTRAN program running in FORTRAN IV+ on RSX-11M:

```
          INTEGER*4 COUNT
          INTEGER*2 JULTES,JULIAN,DAY,MONTH,YEAR,JULTEM,HALF
!
          DO 1099 COUNT=0,32767
            JULTES=COUNT
            CALL UNJUL(JULTES,HALF,DAY,MONTH,YEAR)
            JULTEM=JULIAN(DAY,MONTH,YEAR)+HALF
            IF (JULTEM.EQ.JULTES) GOTO 1099
            TYPE 10,JULTES,JULTEM,HALF,DAY,MONTH,YEAR
10          FORMAT (X, 'Error!',6I7)
1099      CONTINUE
          END
!
! INTEGER FUNCTION TO CONVERT DAY, MONTH AND YEAR TO JULIAN HALF-DAY
!
          INTEGER*2 FUNCTION JULIAN(DAY,MONTH,YEAR)
          INTEGER*2 DAY,MONTH,YEAR
!
          JULIAN = (3055*(MONTH+2)/100-(MONTH+10)/13*2-91
        & +(1-(YEAR-YEAR/4*4+3)/4)*(MONTH+10)/13+DAY-1
        & +(YEAR-1977)*365+(YEAR-1977)/4)*2
          RETURN
          END
!
! SUBROUTINE TO CONVERT JULIAN HALF-DAY TO DAY, MONTH AND YEAR
!
          SUBROUTINE UNJUL(JULIAN,HALF,DAY,MONTH,YEAR)
          INTEGER*2 JULIAN,HALF,DAY,MONTH,YEAR,TEMP1,TEMP2
!
          HALF = JULIAN/2
          TEMP1 = HALF/1461
          TEMP2 = HALF-TEMP1
          YEAR = TEMP2/365
          IF (TEMP2/1460*1460.EQ.TEMP2.AND.(HALF+1)/1460.GT.TEMP1)
        & YEAR = YEAR-1
          TEMP1 = TEMP2-(YEAR*365)+1
          YEAR=YEAR+1977
          TEMP2 = 0
          IF (YEAR/4*4.EQ.YEAR) TEMP2 = 1
          DAY = TEMP1
          IF (TEMP1.GT.59+TEMP2) DAY = DAY+2-TEMP2
          MONTH = (DAY+91)*100/3055
          DAY = DAY+91-MONTH*3055/100
```

```
MONTH = MONTH-2
TEMP1 = 0
IF (HALF*2.NE.JULIAN) TEMP1 = 1
HALF = TEMP1
RETURN
END
```

APPENDIX I

DMC DEVICE COUNTERS

The following counters are the only ones applicable to the DMC device.

Number	Standard Text	
1000	Bytes received	
1001	Bytes sent	
1010	Data blocks received	
1011	Data blocks sent	
1020	Data errors inbound	
	0	NAKs sent, header block check error
	1	NAKs sent, data field block check error
1021	Data errors outbound	
1030	Remote reply timeouts	
1031	Local reply timeouts	
1041	Local buffer errors	
	0	NAKs sent, buffer unavailable

None of the other standard counters can be kept due to the nature of the DMC hardware. The "Data errors outbound" counter is kept with no bitmap. It represents the sum of all NAKs received.

Since the counters kept by the DMC firmware cannot be zeroed in the way that driver-kept counters can, the recommended technique for providing the zero capability is to copy the base table counters when a zero is requested. The numbers returned when counters are requested are then the difference between the saved counters and the current base table.

APPENDIX J

GLOSSARY

NOTE

Terms that derive from other related specifications (such as hops, cost, delay, etc.) are defined in those specifications.

active areas

Active areas are known areas which are currently reachable.

active circuits

Active circuits are known circuits in the ON or SERVICE state.

active lines

Active lines are known lines in the ON or SERVICE state.

active logging

Active logging describes all known sink types that are in the ON or HOLD state.

active nodes

All reachable nodes as perceived from the executor node are active nodes. (See Section 3.1)

adjacent node

A node removed from the executor node by a single physical line.

characteristics

Parameters that are generally static values in volatile memory or permanent values in a permanent data base. A Network Management information type. Characteristics can be set or defined.

circuit

A logical communications path providing a communications connection between adjacent nodes. A circuit may be identical to a physical link, multiplexed with many other circuits, and/or traffic split over multiple physical links. (See Section 3.4)

circuit level loopback

Testing a specific data link circuit by sending a repeated message directly to the data link layer and over a wire to a device that returns the message to the source.

cleared state

Applied to a line: a state where space is reserved for line data bases, but none of them is present.

command node

The node where an NCP command originates.

controller

The part of a device identification that denotes the control hardware for a line. For a multiline device that controller is responsible for one or more units.

counters

Error and performance statistics. A Network Management information type.

entity

AREA, CIRCUIT, LINE, LOGGING, MODULE or NODE. These are the major Network Management keywords. Each one has parameters with options, and most have specified counters. There are also plural entities, such as, KNOWN LINES, ACTIVE LOGGING, SIGNIFICANT NODES, etc.

executor node

The node in which the active Local Network Management Function is running (that is, the node actually executing the command); the active network node physically connected to one end of a line or circuit being used for a load, dump, or line loop test.

filter

A set of flags for an event class that indicates whether or not each event type in that class is to be recorded.

global filter

A filter that applies to all entities within an event class.

hold state

Applied to logging. A state where the sink is temporarily unavailable and events for it should be queued.

host node

The node that provides services for another node (for example, during a down-line task load).

information type

One of CHARACTERISTICS, COUNTERS, EVENTS, STATUS or SUMMARY. Used in the SHOW command to control the type of information returned. Each entity parameter and counter is associated with one or more information types.

known circuits

All circuits addressable by Network Management in the appropriate data base (volatile or permanent) on the executor node. They may not all be in a usable state.

known lines

All lines addressable by Network Management in the appropriate data base (volatile or permanent) on the executor node. They may not all be in a usable state.

known logging

All logging sink-types addressable by Network Management in the appropriate data base.

known nodes

All nodes with address 1 to maximum address that are either reachable or have a node name plus all names that map to a circuit line.

line

A physical communications path. Line is a Network Management entity.

line level loopback

Testing a specific physical link by sending a repeated message directly to the physical link layer and over a wire to a device that returns the message to the source.

logging

Recording information from an occurrence that has potential significance in the operation and/or maintenance of the network in a potentially permanent form where it can be accessed by persons and/or programs to aid them in making real-time or long-term decisions.

logging console

A logging sink that is to receive a human-readable record of events, for example, a terminal or printer.

logging event type

The identification of a particular type of event, such as line restarted or node down.

logging file

A logging sink that is to receive a machine-readable record of events for later retrieval.

logging identification

The sink type associated with the logging entity (file, console or monitor).

logging sink

A place that a copy of an event is to be recorded.

logging sink flags

A set of flags in an event record that indicate the sinks on which the event is to be recorded.

logging sink node

The node from which logging information comes.

logging source process

The process that recognized an event.

logical link

A connection between two nodes that is established and controlled by the Session Control, End Communication, and Routing layers.

loopback node

A special name for a node, that is associated with a line for loop testing purposes. The SET NODE LINE command sets the loopback node name.

module

A module is a component that does not fit into other entity classifications.

monitor

An event sink that is to receive a machine-readable record of events for possible real-time decision making.

node

An implementation that supports Routing, End Communication, and Session Control. Node is a Network Management entity.

node address

The required unique numeric identification of a specific node.

node identification

Either a node name or a node address. In some cases an address must be used as a node identification. In some cases a name must be used as a node identification.

node name

An optional alphanumeric identification associated with a node address in a strict one-to-one mapping. No name may be used more than once in a node. The node name must contain at least one letter.

node level loopback

Testing a logical link using repeated messages that flow with normal data traffic through the Session Control, End Communication, and Routing layers within one node or from one node to another and back. In some cases node level loopback involves using a loopback node name associated with a particular line.

off state

Applied to a node: a state where it will no longer process network traffic. Applied to a line or circuit: a state where the line is unavailable for any kind of traffic. Applied to logging: a state where the sink is not available, and any events for it should be discarded.

on state

Applied to a node: a state of normal network operation. Applied to a line or circuit: a state of availability for normal usage. Applied to logging: a state where a sink is available for receiving events.

processed event

An event after local processing, in final form.

raw event

An event as recorded by the source process, incomplete in terms of total information required.

reachable node

A node to which the executor node's Routing believes it has a usable communications path.

remote node

To one node, any other network node.

restricted state

A node state where no new logical links from other nodes are allowed.

service password

The password required to permit triggering of a node's bootstrap ROM.

service slave mode

The mode where the processor is taken over and the adjacent, executor node is in control, typically for execution of a bootstrap program for down-line loading or for up-line dumping.

service state

A line or circuit state where such operations as down-line load, up-line dump, or line loopback are performed. This state allows direct access by Network Management to the line.

shut state

A state where existing logical links or X.25 virtual calls are undisturbed, but new ones are prevented.

significant

A subset of known entities for which there is at least one parameter or counter.

sink

(see logging sink)

specific filter

A filter that applies to a specific entity within an event class and type.

station

A physical termination on a line, having both a hardware and software implementation.

status

Dynamic information relating to entities, such as their state. A Network Management information type. Also, a message indicating whether or not an NCP command succeeded.

substate

An intermediate state that is displayed as a tag on a state display.

summary

An information type meaning most useful information.

target node

The node that receives a memory image during a down-line load, generates an up-line dump, or loops back a test message.

tributary

A station on a multipoint line that is not a control station.

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Did you find errors in this manual? If so, specify the error and the page number.

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) ______

Name ______ Date ______

Organization ______

Street ______

City ______ State ______ Zip Code or Country ______

— — — — —Do Not Tear - Fold Here and Tape — — — — — —

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

SOFTWARE DOCUMENTATION
1925 ANDOVER STREET TW/E07
TEWKSBURY, MASSACHUSETTS 01876

— — — — Do Not Tear - Fold Here and Tape — — — — — —

Cut Along Dotted Line

Printed in U.S.A.