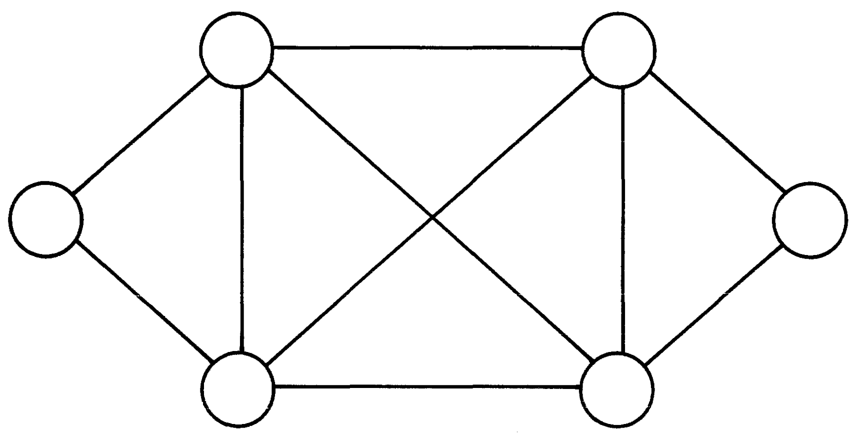
Order No. AA-D600A-TC

NSP



To order additional copies of this document, contact the Software Distribution Center, Digital Equipment Corporation, Maynard, Massachusetts 01754.

digital

DECNET

DIGITAL NETWORK ARCHITECTURE

Network Services Protocol (NSP)

Functional Specification

Version 3.1

March 1978

digital equipment corporation · maynard, massachusetts

First Printing, March 1978

ABSTRACT

This document describes the syntax and semantics of the Network Services Protocol (NSP). NSP creates an interprocess communication mechanism among the nodes of a DECnet network. It is concerned with the set of services provided within the network as well as the management and routing of data within the network. It assumes that the network nodes are connected by error-free physical channels that transmit and receive data blocks in multiples of 8-bit bytes. These error-free channels may be provided by some physical link protocol implemented in hardware and/or software.

This document is intended to serve as an aid for those implementing NSP, as well as for those seeking more general information on the protocol. It is not intended to instruct individuals on the principles of networking or data communications.

Table of Contents

List of Illustrations

List of Tables

1.0 SCOPE

This document describes the Network Services Protocol (NSP), its functions, characteristics and operational capabilities. Sections 1.0 through 6.0 provide the reader with a general understanding of the structure and principles of NSP. Appendix A is a glossary of NSP Terms. Appendices B through G provide more detailed information on NSP. They are presented as an aid to assist those that will implement NSP.

This document describes the Network Services Protocol as specified for Phase II DECnet. Implementations of NSP conforming to this specification should interoperate with Phase II DECnet products (i.e., assuming compatible options, parameters and the appropriate network topology is employed).

Four terms utilized within this document either conflict with the common industry standard definition or have more than one definition. These are NSP, node, message and segment.

The term "NSP" has two usages within this document. It is employed to describe both the protocol, its structure and place within the DIGITAL Network Architecture (DNA) and the process (i.e., a software or hardware module) that supports the protocol.

The word "node" as used in this document refers to an implementation of NSP. Typically, an NSP implementation resides in a computer. However, it is possible for multiple nodes to exist within a single computer.

The word "message" refers to either dialogue message or NSP message. A dialogue message is the unit of information the operating system or dialogue process wishes to send over a logical link to another operating system or dialogue process. An NSP message is a unit of information sent by one implementation of NSP to another.

The term "Segment" refers to either a dialogue segment or an NSP segment. A dialogue segment represents a portion of a dialogue message. An NSP segment is an NSP message that is numbered.

2.0 FUNCTIONAL DESCRIPTION

The Network Services Protocol (NSP) was developed specifically for DECnet. As part of the DIGITAL Network Architecture (DNA), NSP has the primary responsibility for establishing communication links between different processes, routing messages, and managing various network activities.

2.1 Relationship to DECnet

DECnet is a family of hardware and software products that create distributed networks from DIGITAL computers and their interconnecting data links. DECnet creates a general mechanism for sharing resources and providing interprogram communications within a distributed data processing environment. DECnet implementations adhere to a common network architecture that defines the structure and protocols that each must use to communicate through the network. The DIGITAL Network Architecture (DNA) defines this common structure.

DNA provides a modular design for DECnet. Its functional components are defined within three distinct layers:

1. The Physical Link Control Layer (which provides the management of communications over a physical link).

2. The Network Services Layer (which routes messages between source and destination nodes and manages logical data channels).

3. The Application or Dialogue Layer (which supports user services and programs and provides remote I/O device and file access).

The Network Services Protocol was designed to meet the functional requirements of the Network Service Layer.

2.2 NSP Logical Link Service

NSP provides the facility for two dialogue processes (typically, user-written programs residing at different nodes) to exchange information regardless of their physical location within a network. This facility is referred to as the Logical Link Service.

Logical Link Service allows a dialogue process to establish a connection (called a logical link) to another dialogue process. Once the logical link is established, each dialogue process may send data to the other dialogue process via the logical link. When the dialogue is complete, either dialogue process may request that the logical link be disconnected. A logical link provides both guaranteed delivery (i.e., delivery of information to an area of storage accessible to the destination dialogue process) and sequentiality.

2.3 Summary of Functions

A list of the major NSP functions is provided below. Section 2.4 presents a model of the NSP functional organization. A description of NSP operation is provided in Sections 5.0, and 6.0.

1. Operation of a logical link at the dialogue level:

    a. Creation of logical links.

    b. Transmission and reception of data over logical links.

    c. Transmission and reception of interrupt data over logical links.

    d. Destruction of logical links.

    e. Provision for logical link flow control.

    f. Assurance of guaranteed delivery and guaranteed sequentiality.

    g. Provision for authorization of the use of logical links.

2. Routing in a network:

    a. Allow nodes to communicate with adjacent nodes.

    b. Allow satellites of a star topology to communicate with each other (subject to certain restrictions).

    c. Provides for communication in more complex topologies in the future.

3. Network Management/Security:

    a. Detects loss of communications and node restarts.

    b. Provision for a node verification feature at start-up (i.e. when initializing with a neighbor).

2.4 Functional Organization

All NSP functions may be grouped into two areas: (1) those that support logical link operation; (2) those that support network management. A brief description of the major functional components found in each area is provided below.

2.4.1 Logical Link Facility - NSP has the responsibility for establishing a logical link between two dialogue processes. This is accomplished via a well-defined set of control messages sent back and forth between NSP modules. Once a logical link has been established, data may be exchanged over the link. Information sent over the link can be (1) Normal data (i.e., data segments or messages); (2) Interrupt data or (3) Link Service Messages. Dialogue messages are usually sent as normal data segments over the link. Unless the message to be sent is small enough to be transmitted through the network intact, NSP may break up the message into segments and transmit each segment individually. When a small amount of information is to be sent that is not sequentially related to normal data and is of a high priority to the sender (e.g., alarm condition data), then interrupts may be used. Link Service Messages are sent primarily to control the flow of data sent over the logical link.

To ensure the integrity and cohesiveness of the information being exchanged and to allow a receiving NSP module to discard received segments, each NSP module employs a segment acknowledgment scheme. This scheme keeps track of the data segments sent (numbered by NSP) and ascertains whether or not retransmission is necessary. At any time during a dialogue exchange, NSP will allow either party to abort or terminate the conversation. When NSP has been properly notified to do so, it will disassemble or destroy the logical link connection.

2.4.2 Network Management - NSP interfaces directly with the physical link control layer which detects errors in communications with adjacent nodes. These errors may be due to a failure of either the physical link to an adjacent node or the adjacent node itself. NSP is informed of these errors and may decide to stop using a particular physical link because of them.

When a node is coming back into service or is being added to an existing network, NSP provides initialization assistance. In addition, NSP has been designed so that some nodes may be able to perform some routing or logical link services on behalf of other nodes.

2.5 The NSP Environment

NSP Version 3.1 was designed for Phase II DECnet products. These products will interoperate with each other and are intended to interoperate with future DECnet products. Any node implementing this specification will be able to communicate correctly with any other node implementing it provided:

a. they are directly connected via a single physical link, or

b. there is a single intervening node between them and the intervening node contains a Phase II intercept function.

The only valid topologies that may be used with Phase II DECnet products are point-to-point and star topologies. A Phase II implementation was designed to interoperate with another Phase II implementation or with future DECnet products in more complicated topologies provided that the Phase II implementation is adjacent to one and only one intercept node (such an intercept node would also be a future product).

The way in which a Phase II implementation chooses the physical link to communicate with another node is described in Section 6.1. The way in which a Phase II intercept node operates is described in Appendix F.

2.6 Network Communications Model

This section describes those components that are considered fundamental to NSP operation. Section 2.6.1 describes how dialogue messages are handled by NSP for transmission through the network. Section 2.6.2 describes a typical handshaking procedure between two implementations of NSP in order to establish a logical link between two dialogue processes. Section 2.6.3 presents a generic model, to illustrate how NSP sends data over a logical link.

2.6.1 Dialogue Message Transmission - A dialogue message is a unit of information that has meaning to the dialogue processes communicating over a logical link. Because of network constraints (e.g., available buffer sizes, and transmission error probability) the dialogue message may not be able to be sent all in one piece. However, NSP implementations that support segmentation have the capability of taking a large unit of information and breaking it into smaller units (segments) for transmission throughout the network, and delivering the unit of information reassembled to a destination dialogue process. These implementations will reconstruct the original block of data prior to delivery.

2.6.2 NSP Handshaking Sequence - Figure 2-1 depicts the sequence of events that occurs when one dialogue process wishes to send another process a message through the network. Initially, both NSPs exchange control messages to establish a logical link connection. Once this is accomplished, the sending NSP may break up the dialogue message into segments. The sending NSP requires an Acknowledgment (ACK) either explicitly or implicitly from the receiving NSP for each segment transmitted. If a segment was not properly received and a Negative Acknowledgment (NAK) was returned by the receiving NSP, then the sending NSP will retransmit the segment.

A description of the NSP messages and logical link states is provided in Sections 4.0 and 5.3, respectively.

2.6.3 Data Exchange - There are two data streams on a logical link. One stream contains interrupt and link service messages; the other contains segments of data messages. Each stream may be considered a subchannel on the logical link. On the interrupt and link service subchannel (INT/LS) all messages are single NSP segments. On the DATA subchannel however, messages may be broken up by NSP (segmentation) for transmission through the network. NSP segment acknowledgment is performed independently for each subchannel.

2.6.3.1 Segmentation Parameters - If NSP is to automatically break dialogue messages up into segments, it must know how. The transmit segment size parameter (ascertained when a logical link is established) limits the transmitter to a value that is specified when the logical link is established.

NODE A: Dialogue Process | Sending NSP

NODE B: Receiving NSP | Dialogue Process

1. Dialogue Process at Node A requests connection.

   [CONNECT INITIATE] →

   Logical Link in CIS State

2. NSP at Node B receives the Connect Initiate Message.

   [CONNECT INITIATE]

   Logical Link in CIR State

3. Dialogue Process at Node B accepts connection.

   ← [CONNECT CONFIRM]

   Logical Link in RUN State

4. NSP at Node A receives the Connect Confirm Message.

   [CONNECT CONFIRM]

   Logical Link in the RUN State

5. Dialogue process at Node A requests that a message (that is two segments long) be transmitted. NSP at Node A sends the first data segment.

   [DATA] →

   Logical Link in RUN State

6. NSP at Node B acknowledges receipt of the first data segment.

   ← [ACK]

   Logical Link in RUN State

7. NSP at Node A sends the second data segment.

   [DATA] →

   Logical Link in RUN State

8. NSP at Node B acknowledges receipt of the second data segment and gives the dialogue message to the dialogue process at Node B.

   ← [ACK]

   Logical Link in RUN State

9. Dialogue Process at Node A requests a disconnection and NSP at Node A sends a Disconnect Initiate Message.

   [DISCONNECT INITIATE] →

   Logical Link in DIS State

10. NSP at Node B receives Disconnect Initiate Message, sends a Disconnect Confirm Message and informs the dialogue process at Node B.

    ← [DISCONNECT CONFIRM]

    Logical Link is gone

11. NSP at Node A receives Disconnect Confirm Message.

    [DISCONNECT CONFIRM]

    Logical Link is gone.

Figure 2-1 A Typical Handshaking Sequence Between Two Implementations of NSP to Establish a Logical Link Between Two Dialogue Processes

In addition, the receiver maintains two logical link parameters to provide NSP segment acknowledgment:

1. The INT/LS subchannel receive number

2. The DATA subchannel receive number

When a logical link connection is established between two dialogue processes, these parameters are set to 0. As NSP segments are transmitted and positively acknowledged over the logical link, the appropriate receive number will be incremented.

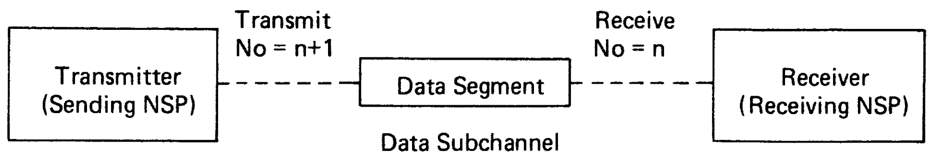
The transmitter will also maintain two parameters to control the operation of NSP segment acknowledgment:

1. The INT/LS subchannel transmit number

2. The DATA subchannel transmit number

These parameters reflect the value of the NSP segment numbers to be assigned on the INT/LS or DATA subchannel. When a logical link connection is established, the value of these parameters is 1.

2.6.3.2 Typical Operation - Figure 2-2 depicts the concept of NSP segmentation with segment acknowledgment. A typical operation is provided for both transmitter and receiver.



Transmitter Operation:

Transmitter assigns the NSP data segment an NSP Segment Number that is equal to the value of the transmit number parameter for the subchannel.

If a NAK is received, the transmitter will retransmit the data segment.

If an ACK is received, the transmitter will then assign the next sequential number to a data segment waiting to be sent.

NOTE

Actually, the transmitter can send more than one segment before receiving an ACK. The number of outstanding (i.e., acknowledged) segments that NSP will allow on a given subchannel is implementation-dependent.

Receiver Operation:

Receiver will process NSP segments on a subchannel in order (i.e., according to the NSP segment number).

If an NSP segment number is less than or equal to the value of the receive number parameter for the subchannel, modulo 4096, it will be discarded.

If the segment number is greater than the receive parameter + 1, modulo 4096, it may be discarded or held until the preceding segments are received.

The received NSP segment must be ACK'd or NAK'd within a specified period or time (3 seconds).

If the received NSP segment is accepted, an ACK is sent back to the transmitter and the receive number will be incremented.

If the received NSP segment is rejected, a NAK is sent back to the transmitter requesting retransmission, and the receive number is not incremented.

Figure 2-2 Operation of Segment Acknowledgment

3.0 INTERFACES

NSP has two primary interfaces: one directly interfaces with the dialogue processes in the DNA Application Layer; the second interfaces with the Physical Link Control Layer. In the sections that follow, several models are provided to describe these interfaces.

3.1 Dialogue Level Interface

At the dialogue level, NSP provides one interface to the dialogue level: the logical link interface.

The generic dialogue interface for logical link services has two forms: (1) a message-oriented interface and (2) a segment-oriented interface. The message interface associates a dialogue process buffer with a single dialogue message. A dialogue message may consist of many segments. The segment interface associates a dialogue process buffer with one or more dialogue segments.

Some capabilities in NSP exist primarily to support the message interface form and some capabilities exist primarily to support the segment interface form. A particular implementation of NSP may support one or the other interface form within a node. Likewise, a logical link may be managed on one end by an implementation oriented toward one interface form and may be managed on the other end by an implementation oriented toward the other.

The Logical Link Interface consists of fourteen functions requested of NSP by a dialogue process. The number of separate interface calls that are defined by an NSP implementation to support these functions and the operation of the implementation when receiving an interface call are implementation-dependent. Furthermore, the information passed across the interface is implementation-dependent. Table 3-1 provides a summary of each function, the information sent to NSP by the dialogue process, and the general information sent to the dialogue process by NSP.

The term link identifier refers to the value by which a dialogue process identifies a specific logical link connected to it. If a dialogue process has several logical links established, then a link identifier will be associated with each logical link. That identifier is unique from the point of view of the dialogue process. If NSP supports more than one dialogue process in a particular node, it will retain the identity of each process. The combination of link identifier and dialogue process identity is sufficient to reference a logical link uniquely.

Table 3-1 Logical Link Interface

FUNCTION	INFORMATION SENT TO NSP	INFORMATION RETURNED TO DIALOGUE PROCESS
1. ISSUE CONNECT REQUEST	1. Destination Node Name 2. Destination Process Identification 3. Source Process Identification 4. Link Identifier 5. Data being sent to distant dialogue process 6. Buffer to receive returned data 7. Access Control Information 8. Segment Interface Information used by a message interface implementation to determine how to segment dialogue messages.	1. Success/Failure Code 2. Returned Data 3. Remote Process's Segment Size (Segment Interface)
2. RECEIVE CONNECT	1. Identity of buffer(s) to receive output parameters	1. Source Node Name 2. Source Process Identification 3. Destination Process Identification 4. Received Data Accompanying the connect request 5. Reply Identifier (a value to be returned on connect acceptance or rejection) 6. Access Control Information 7. Remote Process's Segment Size (segment interface)
3. ACCEPT CONNECT	1. Reply Identifier 2. Link Identifier 3. Data to be returned to the calling dialogue process 4. Segment Size (Segment Interface)	
4. REJECT CONNECT	1. Reply Identifier 2. Data to be returned to the calling dialogue process	
5. TRANSMIT (Message Interface)	1. Link Identifier 2. Identity of buffer containing the dialogue message	1. Success or Failure Indication

Table 3-1 (Cont.) Logical Link Interface

FUNCTION	INFORMATION SENT TO NSP	INFORMATION RETURNED TO DIALOGUE PROCESS
6. TRANSMIT (Segment Interface)	1. Link Identifier 2. Identity of buffer containing a segment. 3. Flag to indicate whether the buffer contains the beginning of a message. 4. Flag to indicate whether the buffer contains the end of a message.	1. Success or Failure Indication
7. RECEIVE (Message Interface)	1. Link Identifier 2. Identity of buffer to receive dialogue message	1. Success or Failure Indication 2. Received Message (Limited by buffer capacity)
8. RECEIVE (Segment Interface)	1. Link Identifier 2. Identity of buffer to receive data	1. Success or Failure Indication (which includes beginning-of-message and end-of-message indications) 2. Received data (i.e. one or more segments)
9. TRANSMIT INTERRUPT	1. Link Identifier 2. Data to Accompany interrupt	1. Success or Failure Indication
10. RECEIVE INTERRUPT	1. Link Identifier 2. Identity of buffer to receive data accompanying interrupt	1. Success or Failure Indication 2. Data accompanying interrupt
11. REQUEST DISCONNECT	1. Link Identifier 2. Data to accompany disconnect request	1. Success or Failure Indication
12. RECEIVE DISCONNECT REQUEST	1. Link Identifier 2. Identity of buffer to receive data accompanying disconnect	1. Disconnect Reason 2. Data accompanying disconnect
13. REQUEST LINK ABORT	1. Link Identifier 2. Data to accompany abort request	(This function always completes successfully)
14. RECEIVE ABORT REQUEST	1. Link Identifier 2. Identity of buffer to receive data accompanying abort.	1. Abort Reason 2. Data accompanying abort

3.2 Physical Link Control Level Interface

The primary purpose of the physical link control level is to provide an error-free data communication path between adjacent nodes. To accomplish this, the physical link control level may use any one of a number of techniques and/or protocols (such as DDCMP) to satisfy its operating requirements.

NSP permits only one physical link between adjacent nodes in a network. If it becomes necessary to install two hardware channels between adjacent nodes, then NSP assumes that the physical link control layer will handle the two channels in a manner that will create a single physical link for NSP.

3.2.1 Operating Requirements

3.2.1 Operating Requirements - The NSP requirements for the physical link control level are as follows:

1. To create an error-free data path. Data must be transferred from one end of a physical link to the other while data integrity is maintained. If data integrity cannot be maintained, then no data should be transferred at all.

2. To transfer NSP messages in proper sequence. Messages should be delivered from one node to the next in the same order as they are sent.

   NOTE

   The sequencing here is up to the physical link control layer. It has nothing to do with the sequence numbers provided by NSP.

3. To manage the characteristics of the hardware channel. If the channel requires management for transmission requests, the physical link control level is responsible for that management. This would be true of half-duplex and multipoint channels.

4. To manage modem and interface control signals. The physical link control level is responsible for controlling all the modem and interface signals necessary for transmission and reception of data. It may do this either directly or via a hardware device driver.

5. To access data in blocks consisting of byte quantities. The interface should accept data in blocks consisting of bytes. Block sizes of at least 192 bytes containing any of the 256 8-bit combinations must be both acceptable for transmission and transparent to the physical link level.

6. To provide restart or initialization notification. If the other end of the link resets or initializes, the physical link control level should pass this information across the interface.

7. To provide start and stop control. NSP should be able to start and stop the operation at the physical link control level.

8. To provide notification of channel error. When a persistent error is detected or a threshold error counter is exceeded, NSP should be notified of this condition. Typical errors might include too many bit errors, line outages, and modem failure.

In addition, the physical link control level may provide a maintenance mode of operation for use in basic diagnostic and bootstrapping functions.

3.2.2 Commands and Notifications - The NSP interface consists of a number of control and data transfer commands being issued and a number of notifications being sent. The actual implementations determine the characteristics and operational details of the commands and notifications. The generic operations and the information transferred across the interface are summarized below.

3.2.2.1 Commands - The following commands are sent to the physical link control layer:

1. Start Link - This command is used to begin initialization of the physical link.

2. Stop Link - This command is used to halt all operations at the physical link control level. If switched communications are used (e.g., dial-up), the connection is not broken.

3. Disconnect Link - This command is used to stop all operation at the physical link control level. If switched communications are used, the connection is broken. This command is equivalent to "stop link" for non-switched facilities.

4. Transmit Message - This command is used to transmit the message requested and notify the sender when the message has been successfully received by the adjacent node.

5. Receive Message - This command is used to receive the next message. In some implementations, NSP might supply a buffer with this command. In other implementations, NSP might provide a buffer pool that the physical link control level would use for buffers and then notify NSP when the message has been placed in the buffer.

3.2.2.2 Notifications - The following notifications are sent to NSP by the physical link control layer:

1. Link Initialized (or Reinitialized) - Notification is given to NSP when the other end of the link has initialized or has entered the maintenance mode (if the physical link control layer suppots a maintenance mode).

2. Threshold Error Exceeded - NSP is notified when a threshold error counter has been exceeded. The threshold limit is usually set to a value that will seldom be exceeded unless the link operation has become such that transmission integrity is questionable. It should be noted, however, that the physical link may continue to operate even though this value has been exceeded.

3. Persistent Error - An error has halted operation.

4.0 NSP MESSAGES

NSP allows dialogue processes to exchange data over logical links. Information necessary for the creation and supervision of logical links is exchanged between NSPs. Data messages carry the dialogue level information between processes and control messages carry information between NSP modules. All messages exchanged by NSP may pass through several intermediate nodes or may pass through a single physical link when nodes are adjacent.

4.1 Message Format Notation

The following notation is used to describe the messages contained herein:

FIELD (LENGTH) : CODING = description of field

where:

FIELD = the name of the field being described

LENGTH = the length of the field as:

1. a number meaning number of 8-bit bytes (octets)

2. a number followed by a "B" meaning number of bits

3. the letters "EX-n" means extensible field, with n being a number that specifies the maximum length of 8-bit bytes in the protocol before interpretation, as described below. If no number is specified, the current maximum length is 1 byte. Extensible fields are variable in length consisting of 8-bit bytes, where the high-order bit of each byte indicates whether the next byte is part of the same field. A 1 means the next byte is part of this field, while a 0 will indicate it is the last byte. The low-order 7-bits of each byte are used as information bits. Extensible fields can be binary or bit map. If they are binary, then 7-bits from each byte are concatenated into a single binary field. If they are bit map, then 7-bits from each byte are used independently or in groups as information bits.

   Note: The bit definitions define the information bits after removing the extension bits and compressing the bytes.

4. the letters "I-n" means this is an image field, with n being a number that specifies the maximum length of 8-bit bytes in the image. The image is preceded by a 1-byte count of the length of the remainder of the field. Image fields are variable in 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.

CODING = the representation type used:

- A = 7-bit ASCII
- B = Binary
- BM = bit map (each bit or group of bits has independent meaning)
- C = constant
- null = interpretation data dependent

Notes:

1. If both the length and coding are omitted, the field represents a generic field with a number of subfields specified in the description.

2. Any bit or field that is stated to be "reserved" must be zero unless otherwise specified.

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 left-most character is in the low-order octet.

5. Bits are numbered with bit 0 on the right (low-order, least-significant bit) and bit 7 on the left (high-order, most-significant bit). For convenience, when the graphic form of a 2-byte field is given, it will be shown converted to a 16-bit word. When a subfield of a message field contains more than one bit, it should be considered a binary value.

6. Unless otherwise specified, the numbers that appear at the top of the message formats represent bit positions.

4.2 Message Formats

4.2.1 General Message Format

4.2.1 General Message Format - In general, NSP Messages have the following format:

RTHDR	MSGFLG	MSGDATA

where:

RTHDR = The RTHDR is used by nodes containing an intercept function to determine the physical link on which to send the message toward the destination node. Once the NSP message has reached its destination, the NSP process will ensure the message is properly delivered.

A Phase II implementation sending a message to a non-adjacent node must include a RTHDR (except for data and acknowledgment messages as defined in Sections 4.1.3 and 4.1.4). A Phase II implementation sending a message to an adjacent node may omit the RTHDR from all messages. This field is optional. Its presence is noted by a "10" in the low-order two bits of the first byte. The route header consists of three fields:

RTFLG	DSTNODE	SRCNODE

where:

RTFLG(EX) : BM = A group of flags used by routing nodes. The format for this field is as follows:

7	6	5	4	3 2	1	0
0	1	0	0	MPRI	1	0

MPRI (Message Priority) is a 2-bit binary subfield. It is set to 1 on transmission and ignored on reception.

DSTNODE(I-6) : A = The destination node name. This is the node name as a character string. This field is limited to digits and uppercase alphabetic characters.

SRCNODE(I-6) : A = The source node name. Its form is the same as DSTNODE (above).

MSGFLG(EX) : BM = A group of fields describing the characteristics of the message. It is identified by "00" in the low-order two bits. The MSGFLG format is:

7	6 4	3 2	1	0
0	SUBTYPE	TYPE	0	0

TYPE(2B) : B = message type (binary)

where:

0 - data message;
1 - acknowledgment message;
2 - control message;
3 - reserved

SUBTYPE(3B) : B = message subtype - used to modify TYPE field.

Type	Subtype (Bits)	Meaning
0	4	0 = normal data segment 1 = Interrupt or link service message
	5	1 = Beginning-of-Message segment (bit 4 = 0)
	6	1 = End-of-Message segment (bit 4 = 0)
	5	1 = interrupt (bit 4 = 1)
	6	reserved = 0 (bit 4 = 1)
1	4	0 = Acknowledges data segment 1 = Acknowledges interrupt or link services message
	5	reserved = 0
	6	reserved = 0
2	4-6	control type (binary): 0 = No operation 1 = Connect initiate 2 = Connect confirm 3 = Disconnect initiate 4 = Disconnect confirm 5 = Startup 6-7 reserved

MSGDATA = data. The remainder of an NSP message.

4.2.2 Data Messages - Three distinct forms of messages may be classified as data messages: normal data segments, interrupt messages, and link service messages.

The normal data segments carry the normal dialogue level information between processes. Interrupt messages carry small amounts of special information between processes. Link service messages carry NSP control information and are used primarily to control the flow of data and interrupt messages.

4.2.2.1 Normal Data Segment - A normal data segment is of the form:

RTHDR MSGFLG	DESTINATION ADDRESS	SOURCE ADDRESS	ACK NUMBER	SEGMENT NUMBER	DATA

RTHDR = Refer to Section 4.2.1.

MSGFLG(EX) : BM = Message identifier. The format for this field is:

7	6	5	4	3	2	1	0
0	EOM	BOM	0	0	0	0	0

where:

EOM(1B) : BM = 1 - denotes segment is end of message.
BOM(1B) : BM = 1 - denotes segment is beginning of message

The combinations are:

EOM=0,BOM=0: a middle segment of a multi-segment dialogue message
EOM=0,BOM=1: the first segment of a multi-segment dialogue message
EOM=1,BOM=0: the last segment of a multi-segment dialogue message
EOM=1,BOM=1: the only segment of a dialogue message

DSTADDR(2) : B = the logical link destination address for the message. This address is assigned when a link is established (i.e., during the NSP connection procedure).

SRCADDR(2) : B = the logical link source address. This address is assigned when a link is established (i.e., during the NSP connection procedure).

ACKNUM(2) : BM = the number of the last NSP data segment successfully received and an ACK or NAK indication. This field is optional. Its presence is indicated by bit 15 being set. The format for this field is as follows:

15	14 12	11 0
1	QUAL	NUMBER

where:

QUAL(3B) : B = message qualifier

0 - ACK
1 - NAK
2-7 - reserved

NUMBER(12B) :B = the segment number.

SEGNUM(2) : BM = The number of this segment, modulo 4096. The format for this field is as follows:

15 12	11 0
0	NUMBER

DATA = the data the dialogue process wishes to send over a logical link. This information will be totally transparent and may use all 8-bits of each byte. Data messages are limited to the maximum message segment size (SEGSIZE) allowed on the logical link in the direction that the message is sent (i.e., SEGSIZE may be different for each direction of each logical link terminating in a node). The length of the data field is ascertained from the total length of the normal data segment and consists of all bytes in the segment after the SEGNUM field. The data field may be null.

4.2.2.2 Interrupt Message (INT) - The Interrupt Message format is as follows:

RTHDR	MSGFLG	DSTADDR	SRCADDR	ACKNUM	SEGNUM	DATA

RTHDR = Refer to Section 4.2.1.

MSGFLG (EX) : BM = Message identifier. The format of the field is:

7	6	5	4	3	2	1	0
0	0	1	1	0	0	0	0

DSTADDR(2) : B = the logical link destination address for the message.

SRCADDR(2) : B = the logical link source address.

ACKNUM(2) : BM = the number of the last NSP interrupt or link service message successfully received and an ACK or NAK indication. This field is optional. Its presence is indicated by bit 15 being set. The format for this field is as follows:

15	14 12	11 0
1	QUAL	NUMBER

where:

QUAL(3B) : B = message qualifier
0 - ACK
1 - NAK
2-7 - reserved

NUMBER(12B) : B = the segment number

SEGNUM(2) : BM = the number of this interrupt message. Numbers for interrupt/link service messages will have no relationship to the numbers assigned to normal data messages. Each message type utilizes a different subchannel on a logical link.

The format for this field is as follows:

15 — 12	11 — 0
0	NUMBER

DATA = the data to be sent over a logical link. This field is totally transparent and may use all 8-bits of each byte. The length of the data field is ascertained from the total length of the interrupt message and consists of all bytes in the message after the SEGNUM field. Interrupt messages are limited to 16 bytes.

4.2.2.3 Link Service Message - The link service message format is:

RTHDR	MSGFLG	DSTADDR	SRCADDR	ACKNUM	SEGNUM	LSFLAGS	FCVAL

RTHDR = Refer to Section 4.2.1.

MSGFLG(EX) : BM = Message identifier. The format for this field is as follows:

7	6	5	4	3	2	1	0
0	0	0	1	0	0	0	0

DSTADDR(2) : B = the logical link destination address for this message.

SRCADDR(2) : B = the logical link source address.

ACKNUM(2) : BM = the number of the last NSP interrupt or link service message successfully received and an ACK or NAK indication. This field is optional. Its presence is indicated by bit 15 being set. The format for this field is as follows:

15	14 — 12	11 — 0
1	QUAL	NUMBER

QUAL(3B) : B = message qualifier

0 - ACK
1 - NAK
2-7 - reserved

NUMBER(12B) : B = the segment number

SEGNUM(2) : BM = the number of this link service message. Numbers for interrupt/link service messages will have no relationship to the numbers assigned to normal data messages. Each message type utilizes a different subchannel on a logical link.

The format for this field is as follows:

15 12	11 0
0	NUMBER

LSFLAGS(EX) : BM = Link service flags. The format for this field is as follows:

7	6	5	4	3 2	1 0
0	0	0	0	FCVAL INT	FC MOD

where:

FCVAL INT(2B) : B = interpretation of FCVAL field

0 - data segment or message request count
1 - interrupt request count
2-3 - reserved

FC MOD(2B) : B = flow control modification

0 - no change
1 - stop data
2 - start data
3 - reserved

FCVAL(1) : B = The number of dialogue messages, normal data segments, or interrupt messages that the sender of the message can receive in addition to those previously requested by a link services message. This number is added to the request count which is maintained by NSP, to determine how many dialogue messages, normal data segments, or interrupt messages will be transmitted via a logical link.

Notes:

1. The transmit request count for segment flow control may be negative. (Negative values are presented in 2's compliment form in the FCVAL field.)
2. If FCVAL is for dialogue or interrupt message flow control, the count must be positive. Use 0 if there is to be no change in the count.

4.2.3 Acknowledgment Messages - Acknowledgment messages are used to ACK (Positively Acknowledge) or NAK (Negatively Acknowledge) normal data segments as well as interrupt and link service messages.

4.2.3.1 Acknowledgment of Normal Data Segment - This acknowledgment has the following form:

RTHDR	MSGFLG	DSTADDR	SRCADDR	ACKNUM

RTHDR = Refer to Section 4.2.1.

MSGFLG(EX) : BM = message identifier. The format for this field is as follows:

7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0

DSTADDR(2) : B = The logical link destination address for the message.

SRCADDR(2) : B = The logical link source address.

ACKNUM(2) : BM = The number of the last normal data segment successfully received and an ACK or NAK indication. This field is required. It has the following form:

15	14 12	11 0
1	QUAL	NUMBER

where:

QUAL(30) : B = message qualifier

0 - ACK
1 - NAK
2-7 - reserved

NUMBER(12B) : B = the segment number

4.2.3.2 Acknowledgment of Interrupt Message or Link Service Message - This acknowledgment has the following form:

RTHDR	MSGFLG	DSTADDR	SRCADDR	ACKNUM

RTHDR = Refer to Section 4.2.1.

MSGFLG(EX) : BM = message identifier. The format is:

7	6	5	4	3	2	1	0
0	0	0	1	0	1	0	0

DSTADDR(2) : B = The logical link destination address for the message.

SRCADDR(2) : B = The logical link source address.

ACKNUM(2) : BM = The number of the last NSP interrupt or link service message successfully received and an ACK or NAK indication. This field is required. It is of the form:

15	14 12	11 0
1	QUAL	NUMBER

where:

QUAL(30) : B = message qualifier

0 - ACK
1 - NAK
2-7 - reserved

NUMBER(12B) : B = the segment number

4.2.4 Control Messages - Control Messages are used to pass information between NSP modules. These messages are divided into three groups:

1. Control messages for logical link operation;
2. Control messages for node startup/initialization; and
3. Control messages for testing.

4.2.4.1 Control Messages for Logical Link Operation -

4.2.4.1.1 Connect Initiate (CI) - The Connect Initiate Message is used to request a logical link. A Connect Initiate Message has the following form:

RTHDR	MSGFLG	DSTADDR	SRCADDR	SERVICES	INFO	SEGSIZE	DATA-CTL

where:

RTHDR = Refer to Section 4.2.1.

MSGFLG(EX) : BM = message identifier. The format for this field is as follows:

7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	0

DSTADDR(2) : B = the destination logical link address. This address will be 0 to allow the receiving NSP to assign a number dynamically.

SRCADDR(2) : B = the source logical link address. This number is assigned by the sending NSP and will be used by the destination to address all messages for this logical link. The value 0 is illegal.

SERVICES(EX) : BM = requested services. The format for this field is as follows:

7	6	5	4	3 2	1	0
0	0	0	0	FCOPT	0	1

where:

FCOPT(2B) : B = flow control options.

0 - none
1 - segment request count
2 - message request count
3 - reserved

INFO (EX) : BM = information. The format for this field is as follows:

7	6	5	4	3	2	1 0
0	0	0	0	0	0	PRI

where:

PRI (2B) : B = the link priority.

This subfield is set to 1 on transmission and ignored on reception.

SEGSIZE(2) : B = the maximum size (in bytes) of a normal data segment to be received on this logical link.

DATA-CTL = The Connect Initiate Data field. This field has the following form:

/-------------------byte positions-------------------\

0	n	m	m+1	l	k	j
DSTNAME	SRCNAME	MENU	RQSTRID	PASSWRD	ACCOUNT	USRDATA

where:

DSTNAME(*-19) : = Destination process name

SRCNAME(*-19) : = Source process name

MENU(EX) : BM = Field format control:

Bit 0 = 1 RQSTRID, PASSWRD, ACCOUNT fields included
Bit 1 = 1 USRDATA field included
Bit 2-6 = 0 reserved, must be zero

RQSTRID(I-16) : B = source user identification data for access verification

PASSWRD(I-8) : B = access verification password

ACCOUNT(I-16) : B = link or service account data

USRDATA(I-16) : B = end-user data field

4.2.4.1.2 Connect Confirm (CC) - The Connect Confirm Message is used to complete the establishment of a logical link requested via a Connect Initiate Message. The Connect Confirm Message is of the form:

RTHDR	MSGFLG	DSTADDR	SRCADDR	SERVICES	INFO	SEGSIZE	DATA-CTL

where:

RTHDR = Refer to Section 4.2.1.

MSGFLG(EX) : BM = message identifier. This field has the following form:

7	6	5	4	3	2	1	0
0	0	1	0	1	0	0	0

DSTADDR(2) : B = the destination logical link address. This will not be 0. It is the value of the SRCADDR field from the Connect Initiate Message.

SRCADDR(2) : B = the source logical link address. This number is assigned by the sending NSP and will be used to address all messages for this logical link. The value 0 is illegal.

SERVICES(EX) : BM = requested services. This field has the following form:

7	6	5	4	3 2	1	0
0	0	0	0	FCOPT	0	1

where:

FCOPT(2B) : B = flow control options.

0 - none
1 - segment request counts
2 - message request counts
3 - reserved

* See Section 6.2.1.1 for process name formats.

INFO (EX) : BM = information. This field has the following form:

7	6	5	4	3	2	1 0
0	0	0	0	0	0	PRI

where:

PRI (2B) : B = the link priority. This subfield is set to 1 on transmission and ignored on reception.

SEGSIZE(2) : B = the maximum size of the user data message segment to be received on this logical link.

DATA-CTL(I-16): B = User-supplied data.

4.2.4.1.3 Disconnect Initiate (DI) - The Disconnect Initiate Message has the following format:

RTHDR	MSGFLG	DSTADDR	SRCADDR	REASON	DATA-CTL

where:

RTHDR = Refer to Section 4.2.1

MSGFLG(EX) : BM = Message identifier. The format is as follows:

7	6	5	4	3	2	1	0
0	0	1	1	1	0	0	0

DSTADDR(2) : B = The logical link destination address for a message.

SRCADDR(2) : B = The logical link source address.

REASON(2) : B = disconnect reason. See Appendix D.

DATA-CTL(I-16): B = user data.

4.2.4.1.4 Disconnect Confirm (DC) - A disconnect confirm message has the following format:

RTHDR	MSGFLG	DSTADDR	SRCADDR	REASON

where:

RTHDR = Refer to Section 4.2.1.

MSGFLG(EX) : BM = Message identifier. This field has the following form:

7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0

DSTADDR(2) : B = The logical link destination address for this message.

SRCADDR(2) : B = The logical link source address.

REASON(2) : B = disconnect reason. See Appendix D.

4.2.4.2 Control Messages for Node Startup/Initialization -

4.2.4.2.1 Node Initialization - The Node Initialization Message is exchanged by adjacent nodes at NSP initialization or startup on all physical links.

It is used to perform the following:

1. Request a node verification feature;
2. Request functions from the other node;
3. Provide a list of features and version supported;
4. Set some dynamic variables in the nodes; and
5. Determine name and number of adjacent node.

If there is a parameter mismatch or the initialization sequence has not been completed properly, then the initialization process is restarted. If one of the nodes is able to adjust its initialization parameters dynamically (based on the Node Initialization Message that it just received), then it may do so for subsequent Node Initialization Messages that it sends. This message is never sent with a RTHDR. It is only valid from adjacent nodes. The Node Initialization Message has the following form:

MSG FLG	START TYPE	NODE ADDR	NODE NAME	FUNCTIONS	REQUESTS	BLK SIZE	NSP SIZE	MAX LNKS	ROUT VER	COMM VER	SYS VER

where:

MSGFLG (EX) : BM = Message identifier. The format for this field is as follows:

7	6	5	4	3	2	1	0
0	1	0	1	1	0	0	0

STARTTYPE(1): B = Type of startup message.
1 = Node Initialization Message

NODEADDR(EX-2):BM = The source node address. The value of this field must be greater than 1 and less than 241. No two nodes in the same network may have the same node address.

NODENAME(I-6): A = The source node name as in RTHDR. Refer to Section 4.2.1.

NOTE

A node may have only one node address and node name. No two nodes may have the same node address or name.

FUNCTIONS(EX): BM = The functions supported at this node. The format for this field is as follows:

7 3	2 0
0	INT

where:

INT(3B) : B = Intercept functions. This subfield must be set to 0 on transmit by a Phase II non-intercept node and to 7 on transmit by a Phase II intercept node. It may be 0 or 7 on receive.

- 0 - no intercept functions
- 1-6 - reserved
- 7 - intercept functions

REQUESTS(EX) : BM = the requests desired of the receiver by the sender of the initialization message. The format for this field is as follows:

7 3	2 1	0
0	RINT	VERIF

where:

RINT(2B) B = intercept requests. This subfield must be set to 3 on transmit by a Phase II non-intercept node and to 0 on transmit by a Phase II intercept node. It may be 0 or 3 on receive.

- 0 - no intercept requested
- 1 - reserved
- 2 - reserved
- 3 - intercept requested

VERIF(1B) : BM = 1 - Node Verification message required

BLKSIZE(2) : B = the maximum physical block size the link will accept (excluding physical link protocol overhead).

NSPSIZE(2) : B = the maximum NSP message segment size this node will accept. This includes RTHDR, MSGFLG and MSGDATA. This number must be less than or equal to BLKSIZE.

MAXLNKS(2) : B = the maximum number of links this node will support. The value is limited to 4095.

ROUTVER(3) : = the version of the routing part of NSP as 3 binary bytes:

byte 1 - version number;
byte 2 - ECO number; and
byte 3 - customer level number.

This field may be transmitted as either three zero bytes or 3, 1, 0 for bytes 1, 2 and 3, respectively.

COMMVER(3) : = the version of the communications (end-to-end) part of NSP as 3 binary bytes. Byte definition is the same as above (ROUTVER).

SYSVER(I-32) : A = a string describing the operating system, date of creation, etc.

4.2.4.2.2 Node Verification - The Node Verification Message provides a password protection feature for new nodes entering a network. It is sent in response to a request in the Node Initialization Message. The Node Verification Message format is as follows:

MSGFLG	STARTTYPE	PASSWORD

MSGFLG(EX) : BM = Message Identifier. The form is:

7	6	5	4	3	2	1	0
0	1	0	1	1	0	0	0

STARTTYPE(2) : B = Type of Startup message.
2 = Node Verification Message.

PASSWORD(8) : B = the password for the requesting node.

NOTE

Passwords may be: different for each node address in the network, the same for the entire network, or even omitted entirely. This is a network specific option.

4.2.4.3 Control Messages for Testing -

4.2.4.3.1 No Operation (NOP) - The NOP Message is used to generate traffic on physical links that are very lightly used so that link failures may be detected. The NOP format is:

RTHDR	MSGFLG	TSTDATA

RTHDR = Refer to Section 4.2.1.

MSGFLG(EX) : BM = Message Identifier. The format for this field is as follows:

7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0

TSTDATA = Any test data pattern. This message is ignored on receive.

5.0 LOGICAL LINK OPERATION

This section describes the Logical Link Operating Requirements, Link Addressing, the logical link states and properties. The description provided herein is not meant, in any way, to specify the software design of either processing modules or data bases. It does, however, provide a general description of NSP operation and states the minimum requirements for NSP implementations.

When a dialogue process requests a connection to establish a data path to another process an attempt is made to create a logical link. Once established, the logical link becomes the network medium for providing dialogue level services. In order to provide such services, the NSP implementation in one node must be able to exchange messages with the NSP implementation in another node.

5.1 Operating Requirements

5.1.1 Logical Link Requirements - Once a logical link has reached the RUN state, it may be used by a pair of dialogue processes to exchange messages. This can only be accomplished if the NSP implementation in each node cooperates in managing the logical link.

5.1.1.1 A Receiving NSP - An NSP implementation that receives data on a logical link must be able to control the flow of messages containing dialogue process data from the transmitter.

An NSP implementation that receives data on a logical link is also able to NAK a segment containing dialogue process data causing the segment to be retransmitted.

5.1.1.2 A Transmitting NSP - An NSP implementation that transmits data on a logical link may optionally break up a dialogue message for transmission on a logical link.

5.1.2 Rules for Receiving NSP Messages - The following are general rules about receiving NSP messages:

1. If a received NSP message is too long, the excess information is truncated.

2. If bits 4, 5, or 6 of RTFLG contain values other than the defined constant values, the entire message is ignored. If bits 4, 5, or 6 of RTFLG contain the defined constant values, then the message is accepted, even if RTFLG contains an additional byte (i.e., is extended to two bytes). If RTFLG is extended to two bytes, the high order byte is ignored.

3. If either the DSTNODE field or the SRCNODE field contains more than six characters of node name, then the entire message is ignored.

4. If the TYPE or SUBTYPE subfields of the MSGFLG field contain reserved binary values, then the entire message is ignored. If the TYPE and SUBTYPE subfields of the MSGFLG field contain no reserved values, then the message is accepted, even if MSGFLG contains an additional byte (i.e., is extended to two bytes). If MSGFLG is extended to two bytes, the high order byte is ignored.

5.2 Logical Link Addressing

5.2.1 Logical Link Address Defined - Every NSP message with the exception of a Node Initialization Message and NOP message contains logical link addressing information. A particular logical link will have two addresses associated with it, one for each node in which the link terminates. A particular node may not terminate more than 4095 logical links simultaneously.

A logical link address is a 16-bit value used by a node for communicating over a particular logical link. The logical link address has the following properties:

1. It cannot be zero.

2. It must be unique with respect to the logical link addresses of all other logical links terminating in the same node.

3. When a logical link is disconnected, the address associated with that link should not be used again for as long as possible.

   There are 65,535 possible logical link addresses. If there is a maximum of m logical links that may terminate in the node at the same time, (where m is the value of the MAXLNKS field of the Node Initialization Message sent by the node), then the remaining (65,535-m) logical link addresses should be used before reusing the logical link address of the disconnected logical link.

4. If a node is being intercepted by an intercept node, then the low-order n bits of the logical link address must be other than zero and unique with respect to all other logical links terminating in the intercepted node.

   The value, n, is the smallest power of 2 that is greater than the maximum number of links that may terminate in the node at the same.

5.2.2 Requirements - An NSP implementation must maintain the following information in a data base for each logical link that terminates in the node.

1. Local link address - the logical link address by which the implementation identifies a particular logical link.

2. Remote node identity - the identity of the node in which the other end of the logical link terminates.

3. Remote link address - the logical link address by which the NSP implementation in the remote node identifies a particular logical link.

All messages whose reception may cause a logical link state transition contain the remote node identity explicitly or implicitly and the local link address and remote link address explicitly in the DSTADDR and SRCADDR fields, respectively.

When an NSP message is received, the following rules apply:

1. If the NSP message is a Connect Initiate with a non-zero DSTADDR field, it is discarded as a protocol error.

2. If the NSP message is a Connect Initiate with a zero DSTADDR field, then it always matches an IDLE link (see Section 5.3).

3. If the NSP message is not a Connect Initiate then it is an event for an existing logical link if and only if the DSTADDR field and source node identity are equal to the local link address and remote node identity, respectively, for the existing logical link. If a RTHDR field is not present (either because the message is a data or acknowledgment message or because of point-to-point operation), then the source node identity check is bypassed. Note, that these tests are necessary but not sufficient. If there is no existing logical link for which this test succeeds, then the NSP message is an event on an IDLE link.

4. Normally, an NSP message that is not a Connect Initiate is an event on an existing logical link if and only if it passes test 4 above and the SRCADDR field from the message is equal to the remote link address for the existing logical link. A received message with a SRCADDR field equal to zero is normally discarded.

5. If the message is not a Connect Initiate and it contains a zero DSTADDR field, then the message is discarded.

5.3 Logical Link States

A logical link passes through several states between establishment and disconnection. A brief definition of each of the logical link states is provided below.

1. CONNECT INITIATE SENT (CIS)
   A Connect Initiate Message has been sent as the result of a connect request.

2. CONNECT INITIATE RECEIVED (CIR)
   A Connect Initiate Message has been received.

3. RUNNING (RUN)
   A logical link has been established for use by a pair of dialogue processes.

4. DISCONNECT INITIATE SENT (DIS)
   A Disconnect Initiate Message has been sent as the result of one of several events occurring in the node that sent the message, including a disconnect request by the dialogue process using the logical link.

5. IDLE
   The logical link is not in use. This is not an actual state.

Figure 5-1 is the logical link state diagram. For details on operation, refer to the Logical Link State Tables in Appendix B.

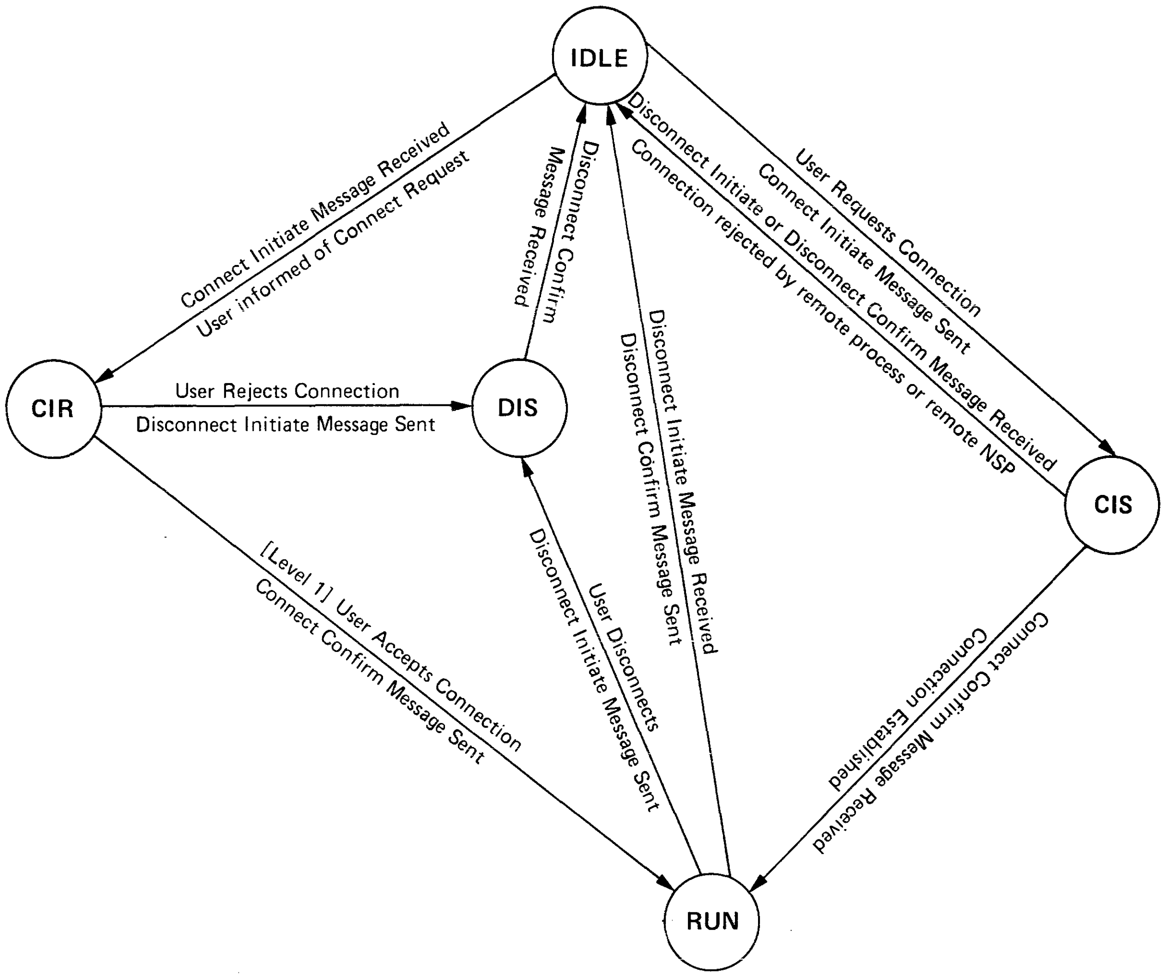


Figure 5-1 Logical Link State Diagram

The following events may have an effect on the state of a logical link:

1. A dialogue process requests a connection.

2. A dialogue process accepts a connection request.

3. A dialogue process rejects a connection request.

4. A dialogue process disconnects or aborts a logical link.

5. A dialogue process aborts while using a logical link.

6. A Connect Initiate message is received.

7. A Connect Confirm message is received.

8. An ACK message is received.

9. A NAK message is received.

10. A DATA message is received (this includes Interrupt and Link Service Messages).

11. A Disconnect Initiate message is received.

12. A Disconnect Confirm message is received.

13. A "no path" event occurs for the node that terminates the other end of the link. Refer to section 6.5.2.

14. A link error event occurs. Refer to section 6.5.1.

5.4 Logical Link Properties

Logical links have four properties used for link management. Three of these properties are always present for a link in the RUN state. These are:

1. Any NSP segments sent by the transmitter must be positively or negatively acknowledged by the receiver.

2. The receiver may stop data messages from being sent by the transmitter, and the receiver may allow data messages to be sent from the transmitter.

3. The receiver may control, by request count, the number of interrupt messages that may be sent by the transmitter.

The fourth, optional property is that the receiver may control, by request count, the number of either segments or dialogue messages which may be sent by the transmitter. This property is selected by the receiver at the time the logical link is established. It is requested via the SERVICES fields of the Connect Initiate and Connect Confirm messages. The receiver in each node that terminates a logical link has an opportunity to request this property, regardless of which node initially requested the connection. Either node may prevent the establishment of the logical link if its transmitter is not capable of supporting the property.

5.5 NSP Segmentation

Normal data segments have a maximum length, but dialogue messages do not. Implementations supporting the message form of the dialogue level interface must break up dialogue messages into segments for transmission and must reassemble segments into dialogue messages on reception. This process is referred to as segmentation.

5.5.1 Segmentation Parameter - To efficiently break up dialogue messages into segments, a single parameter is required by the transmitter for each logical link. This parameter is called the Transmit Segment Size. The value of this parameter is determined by NSP when the logical link is established. The value is equal to the minimum of:

a. The value of the SEGSIZE field from either the received CI message (if the remote dialogue process requested the connection) or the received CC message (if the local dialogue process requested the connection), and

b. The value of the NSPSIZE parameter, for this node minus the maximum length of an NSP header for a data message.

5.5.2 Segment Acknowledgment - NSP segment acknowledgment is performed on both the data subchannel and the interrupt or link service subchannel. The operation of NSP segment acknowledgment is identical for both subchannels in the receiver and nearly identical for both subchannels in the transmitter.

The receiver requires two logical link parameters for the operation of NSP segment acknowledgment:

1. the Interrupt/Link Service subchannel receive number; and

2. the Data subchannel receive number.

These parameters are the values of the last NSP segment received and acknowledged on the Interrupt/Link Service subchannel and the Data subchannel. Both parameters are at 0 when the logical link enters the RUN state.

The transmitter requires four logical link parameters for the operation of NSP segment acknowledgment:

1. the Interrupt/Link Service subchannel transmit number;

2. the Data subchannel transmit number;

3. the Interrupt/Link Service subchannel acknowledgment number; and

4. the Data subchannel acknowledgment number.

These parameters are the values of the NSP segment numbers to be assigned on the Interrupt/Link Service subchannel and the Data subchannel and the largest segment numbers acknowledged on the Interrupt/Link Service subchannel and the Data subchannel, respectively. The initial values of the first two parameters are both 1 and the second two parameters are both 0 when the logical link enters the RUN state.

In the following sections, all NSP segment number arithmetic is performed modulo 4096. A segment number, n, is defined to be greater than a segment number, m, if (n-m) < 2048 (modulo 4096).

5.5.3 Receiver Operation - The operation at the receiver is described with respect to a single subchannel since both subchannels are processed identically by the receiver.

The receiver processes NSP segments received on the subchannel, by NSP segment number. That is, each NSP segment number n must be processed before NSP segment number n+1; consequently, the next NSP segment that must be processed on a subchannel is the one whose NSP segment number is one greater than the value of the receive number parameter for the subchannel.

Basically, the receiver operates on two principles:

1. A received NSP segment must be discarded if its NSP segment number is less than or equal to the value of the receive number parameter for the subchannel. This will occur if the segment had previously been received and accepted.

2. An acknowledgment must be sent whenever an NSP segment is received whether or not the segment is discarded. If the segment is discarded because the segment had previously been received and accepted, then the acknowledgment must be positive.

If an NSP segment that has not previously been accepted is received out of order, it may be rejected or it may be held until the NSP segment preceding it is processed.

The received NSP segment must be accepted or rejected, by a positive or negative acknowledgment within a specified period of time (a network parameter whose default value is 3 seconds). If the received NSP segment is accepted, its number becomes the value of the receive parameter for the subchannel, and a positive acknowledgment may be sent. If the received NSP segment is rejected, a negative acknowledgment must be sent.

Positive acknowledgment of NSP segment n, also positively acknowledges all NSP segments whose number is less than n; consequently, each NSP segment does not have to be individually acknowledged.

Sending a negative acknowledgment containing the segment number n in the ACKNUM field, negatively acknowledges each NSP segment whose number is greater than n and positively acknowledges each NSP segment whose number is less than or equal to n. It is acceptable to send a negative acknowledgment containing the segment number n (provided NSP segment number n has been received) irrespective of whether or not NSP segments with numbers greater than n have been received.

An acknowledgment may be sent to the transmitter by either sending an ACK message or piggybacking the acknowledgment on an NSP segment being sent to the transmitter on the same subchannel. The latter method results in more efficient use of the communications hardware.

5.5.4 Transmitter Operation - When the transmitter is about to send an NSP segment for the first time on a particular subchannel, it assigns the NSP segment an NSP segment number equal to the value of the transmit number. The transmit number parameter is then incremented by one.

Once an NSP segment number has been assigned to the data contained in the NSP segment, the association of the segment number with the data must be maintained until the transmitter receives a positive acknowledgment of the NSP segment. If, for any reason, the data in an NSP segment must be retransmitted, then it must have the same NSP segment number that it did the first time.

All implementations must support positive and negative acknowledgments from the receiver. Since it is possible to receive acknowledgments out of order, if the number in an ACKNUM field of a received NSP message is less than the acknowledgment number for the subchannel, then the ACKNUM field is ignored. If the number in the ACKNUM field is equal to or greater than the acknowledgment number for the subchannel and if the number is less than the transmit number parameter for the subchannel, then the ACKNUM field is processed and becomes the acknowledgment number for the subchannel. (In particular, it is valid to receive and process a positive acknowledgment of segment number n followed by a negative acknowledgment of segment number n). If the number is greater than or equal to the transmit number for the subchannel, then the ACKNUM field is not processed.

When any acknowledgment is received, the segment number value of the ACKNUM field contains the number of the "latest" NSP segment being acknowledged on the subchannel. That is, all NSP segments whose numbers are less than or equal to the segment number in the ACKNUM field have been positively acknowledged.

If an NSP segment has been positively acknowledged, the association of the data in the NSP segment with the number of the NSP segment does not need to be maintained any longer. The data in the NSP segment will never require retransmission.

If an acknowledgment is received from the receiver in which the ACKNUM field indicates a negative acknowledgment, then all NSP segments which have been transmitted on the subchannel but which are not positively acknowledged implicitly in the acknowledgment are negatively acknowledged. However, a segment that has already been negatively acknowledged is not considered to be negatively acknowledged a second time unless it has been retransmitted subsequent to the prior negative acknowledgment. There may be no NSP segments that are negatively acknowledged by a message in which the ACKNUM field indicates a negative acknowledgment. NSP segments on the Interrupt or Link Service subchannel that are negatively acknowledged are always retransmitted immediately. NSP segments on the Data subchannel that are negatively acknowledged are subject to flow control constraints.

NOTE

> Any ACKNUM field must be processed even if the NSP segment in which it is contained will be negatively acknowledged.

5.6 NSP Disconnect/Abort

5.6.1 NSP Disconnection - When a dialogue process using a logical link requests a disconnection, the disconnect request is assumed, by the process, to be synchronous with any previous data transmit requests. If there are unacknowledged NSP segments that have been transmitted on the link, NSP may reject the disconnect request from the dialogue process. If NSP honors the request, then the transmitter must queue the disconnect request until all data from the dialogue process has been sent and acknowledged. When all previous data has been acknowledged, the transmitter may then send a Disconnect Initiate Message. The transition of the logical link from the RUN state to the DIS state does not occur until the Disconnect Initiate Message is actually sent. While the disconnect request is queued the link remains in the RUN state.

When the synchronous disconnect request made by the dialogue process is honored, all pending receive requests by the process for the same link are terminated with an indication of the reason; furthermore, any NSP segments received after the synchronous disconnect request are negatively acknowledged.

A disconnect request from a dialogue process completes successfully if and only if:

1. The transition from the RUN state to the DIS state was caused by sending the Disconnect Initiate message (generated as a result of the disconnect request) and

2. The transition from the DIS state to the IDLE state was a result of the reception of a Disconnect Confirm message with a REASON field indicating that it is a response to a Disconnect Initiate message for an active link.

If this scenario does not obtain, then the dialogue process may be informed that the disconnection has completed abnormally, with a possible loss of data on the logical link. See Appendix E, section E.3.5, for restrictions on the use of synchronous disconnect requests.

NOTE

The receiver of a Disconnect Initiate Message should truncate the image data field if it is longer than the specified limit; moreover, if it is truncated, a Disconnect Confirm Message should be returned with the appropriate REASON code (see Appendix D).

5.6.2 NSP Abort - The dialogue process that is using a logical link may request a link abort. In this case, the operation of the transmitter is identical to the operation resulting from a disconnect request, except that the disconnection occurs immediately rather than after all previous data transmit requests have completed and the requesting dialogue process is always given a successful completion indication.

6.0 NETWORK MANAGEMENT

6.1 Routing

6.1.1 Routing Defined - Routing is the general capability to determine the physical link path on which to send a message destined for a particular node, identified by name. A node implementing this specification can route: to itself, to all nodes that are logically adjacent (i.e., physically adjacent nodes that have the physical link between them in the "ON" state) to it; and (possibly) to one or more non-adjacent nodes. Non-intercept nodes only route messages that are generated internally. Intercept nodes may route received messages.

6.1.2 Routing Parameters - The routing parameters that exist in all nodes are: (1) parameters associated with loopback routing (2) parameters associated with each logically adjacent node, and (3) a single pair of parameters for all non-adjacent nodes.

There are two parameters which permit a node to perform loopback routing: a loop-back state parameter and a loop-back selection parameter. The loop-back state parameter can have the logical values "loop-back" or "no loop-back". It is normally set to "no loop-back." The loop-back selection parameter has the identity of a physical link when the loop-back state parameter is set. If the node is not set for a loop-back, then this parameter is not used.

For each node that is logically adjacent, there are two parameters: a node state parameter and a route selection parameter. The value of the node state parameter is always "adjacent" and the value of the route selection parameter is the identity of the physical link that connects this node to the logically adjacent node.

A single pair of parameters is associated with all remaining nodes in the network. These parameters are the non-adjacent node state parameter and the non-adjacent route selection parameter. The non-adjacent node state parameter will be set to one of two values: "accessible" or "not accessible". The initial value for this parameter is "not accessible".

6.1.3 Routing Operation -

6.1.3.1 Loopback Routing - All nodes must be able to route to themselves. This may be accomplished by either routing internally or routing via a physical link. If the value of the loop-back state parameter is "no loop-back" then loopback routing is accomplished internally. If the value of the loop-back state parameter is "loop back" then all NSP messages destined for self must be sent over a physical link whose identity is contained in the loop-back selection parameter.

Whenever a physical link makes the transition to the "ON" state and its associated state control link parameter has the "loop-back" value, then the loop-back state parameter is set to the "loop-back" value, and the loop-back selection parameter is set to the identity of the physical link. Whenever the physical link leaves the "ON" state, the loop-back state parameter is set to the "no loop-back" value.

6.1.3.2 Adjacent Routing - All nodes must be able to route by name to the nodes logically adjacent to them. Whenever an NSP message needs to be sent to a node whose node state parameter is set to "adjacent", the NSP message is sent on the physical link whose identity is contained in the associated route selection parameter.

Whenever adjacent node initialization occurs successfully, the node state parameter for the logically adjacent node is set to "adjacent" and the route selection parameter for the logically adjacent node is set to the identity of the physical link over which the adjacent node initialization took place. Whenever a physical link to a logically adjacent node undergoes a transition out of the ON state, the node state parameter for the node that was formerly logically adjacent on the physical link is set to "not accessible".

6.1.3.3 Non-Adjacent Routing - If an NSP message must be sent to a node which is not logically adjacent, then the non-adjacent node state parameter is examined. If the value of the parameter is "not accessible", the NSP message is discarded. If the value of the parameter is "accessible", the NSP message is sent on the physical link specified by the non-adjacent route selection parameter.

When adjacent node initialization occurs with a node that supports intercept, the value of the non-adjacent node state parameter is set to "accessible" and the non-adjacent route selection parameter is set to the identity of the physical link to the intercept node. Whenever a physical link leaves the "ON" state, two events may occur:

1. if the node that was adjacent on the physical link was an intercept node, then the value of the non-adjacent node state parameter is set to "not accessible"; and

2. a "no path" event is generated for each logical link associated with the physical link (see Section 6.5.2).

6.1.3.4 Multiple Intercept Nodes - It is possible to physically construct a network in which a Phase II implementation is adjacent to two nodes, each with an intercept function. The topological restrictions and the routing operation of such a network are beyond the scope of this specification.

6.2 Process Identification and Access Control

When establishing a logical link, it is necessary to identify both the destination node and the destination process within the node. The specification for the destination process within the node is known as the process name. The source dialogue process specifies the destination process name at link establishment time. Of the two NSP messages exchanged, only the first message has the destination process name information in it.

In the context of the DECnet architecture, a destination process (or dialogue process) may be a user-written program, a system-supplied service, a component of the DECnet control structure, etc. Each such process which is to be accessible via DECnet must have associated with it either a non-zero object type or a zero object type and an object descriptor. There are no other valid combinations of object type values and descriptors. The object type and descriptor (for object type zero processes) are the process name used to specify a destination.

In addition to simply identifying the destination process, it is often necessary for the process requesting communications (the source process) to pass along additional information for access control at the destination node. Processes that provide services or allow for remote system-control functions must be able to verify the privilege or authority of the source process. This access control information, when required, must be specified when the destination process is specified at logical link establishment.

6.2.1 Process Identification - The process identification defined for use within DECnet is a two-part process name, consisting of a one-byte object type code and an optional object descriptor field. A non-zero object type identifies a generic class of processes or services; a zero object type plus object descriptor specifies a particular, unique process within the object type class. Appendix C of this document contains a list of the object types currently assigned.

6.2.1.1 Process Name Field Formats - The process name fields in the Connect Initiate message have one of three formats, depending on the amount and structure of the descriptive information. These three formats are:

Format 00 = Object Type only

```
Byte    0        1
        +----------------+
Field   !00000000!OBJTYPE!
        +----------------+
```

where:

OBJTYPE (1) : B = the object type of the process. It may not be zero.

Format 01 = Object Type and Descriptor

```
Byte    0        1       2 - n
        +--------+-------+-------+
Field   !00000001!OBJTYPE!DESCRPT!
        +--------+-------+-------+
```

where:

OBJTYPE (1) : B = the object type of the process. It must be zero.

DESCRPT (I-16) : B = the process descriptor. A unique name that qualifies the object type.

Format 02 = Object Type, Group, User, Descriptor

```
Byte    0        1       2       4       6 - n
        +--------+-------+-------+-------+-------+
Field   !00000010!OBJTYPE!GRPCODE!USRCODE!DESCRPT!
        +--------+-------+-------+-------+-------+
```

where:

OBJTYPE (1) : B = the object type of the process. It must be zero.

GRPCODE (2) : B = binary group code.

USRCODE (2) : B = binary user code.

DESCRPT (I-12) : B = the process descriptor. A unique name that qualifies the object type.

Notes:

1. Both the GRPCODE and USRCODE fields are unsigned, 16-bit numbers sent low-order byte first (i.e., bytes 2 and 4 are the low-order bytes).

2. Format codes 3 through 255 are reserved for future extensions.

6.2.1.2 Interpretation of Process Names - Usage of the different process name field formats is related directly to the type of process addressed (object type). However, an object type may be addressed with format 1 in one system and format 2 in another. Application programs must be identified by including a unique name as the object descriptor. For those programs, the format 00 name field is invalid since it does not provide sufficient information. Likewise, formats 01 and 02 are invalid for these objects, if specified with a null descriptor.

The majority of the other process types have been defined to provide services by means of a specific dialogue-level protocol, identified by the object type alone. These processes may be addressed via the format 00 name field, without requiring knowledge of a task name or descriptor.

In some Phase II implementations, a non-zero object type process may not be uniquely identified by its object type. Therefore, multiple logical links to a single copy of a process, identified by a non-zero object type, are not guaranteed. Each copy of such a process, however, is identical to every other copy. A process identified by a zero object type with descriptor is guaranteed to be unique. Multiple copies of such processes are not allowed to be each identified by the same zero object type and descriptor.

The basic rules stated above apply directly to process name field formats 00 and 01. Field format 02, which includes the binary group and user codes, is an extension of format 01 primarily intended for use as a source process descriptor. The same rules and restrictions that apply to field format 01 also apply to field format 02, where the group and user codes are not considered to be part of the descriptor. When format 02 is used for specifying a destination process, the group and user code values may be ignored at the destination node. When format 02 is used to specify a source process, the code values are likely to be significant, but only to the destination process since the receiving NSP does not act on the source name.

While the process naming conventions generally require a unique selection of a process, there is no restriction on the individual processes as to how many different names they may have. It is valid for a single process to be addressed both as a general process (object 0 with a name) and as some other object type (without a descriptor). If this option is supported, each process name (object type and descriptor) must be separately defined.

6.2.2 Access Control and Validation - The basic concept of access control recognizes that not all users of any collection of services are privileged or permitted to use any or all of the services available. There are almost always some requests or operations that must be restricted for either reliability, privacy, accountability, or security reasons.

The approach taken to defining a standard access control mechanism for DECnet is to identify the kernels of information to be used in verifying access or other privileges. This document does not attempt to standardize the exact format or content of the information items, nor to specify the acceptance criteria.

The access control mechanism depends on three generic kernels of data:

1. Requestor ID. A character-coded reference that, in a single-node context, uniquely identifies the person or process requesting service.

2. Password. An arbitrary byte string used for cross-check verification (normally uniquely paired with the Requestor ID or with the service).

3. Accounting Data. A character-coded definition that, when paired with the Requestor ID, identifies a "billing address" for service costs at the destination node.

Each access control information kernel may be passed as an integral part of the Connect Initiate NSP message during logical link establishment. These fields may be handled by the receiving NSP or by the destination dialogue process.

Systems that require control over all service requests (such as time sharing systems), may validate the Connect Initiate before passing it to the dialogue process. They might reject invalid requests without informing the specified destination. Other systems may choose to delegate the validation responsibility to each of the dialogue processes. In either case, the receiving NSP may make the information available to the dialogue process if the information is valid.

6.3 Flow Control

6.3.1 Flow Control Parameters - Four parameters are associated with NSP Flow Control:

1. A data flow control switch which may have one of two logical values - "open" or "closed".

2. An interrupt request count.

3. A data request switch which may have one of three logical values - "none", "segment", or "message". This value is set during link establishment and does not change.

4. A data request count - when the value of the data request switch is "segment" or "message".

When the logical link enters the RUN state, the parameters are:

Data Flow Control Switch - open
Interrupt Request Count - 1
Data Request Switch - as specified by the FCOPT field of the Connect Initiate or Connect Confirm message received during logical link establishment.
Data Request Count (if present) - 0

The maximum value for the Interrupt Request Count and Data Request Count is 127. The minimum value for the Interrupt Request Count is 0. The minimum value for the Data Request Count is 0 if the Data Request Switch parameter has the "message" value and -127 if the Data Request Switch parameter has the "segment" value.

6.3.2 Receiver Operation - The receiver operation for flow control is described herein for both interrupt and data messages. Flow control operation is considered a level above the NSP segment acknowledgment scheme previously described. The purpose of flow control operation is to determine whether or not there should be a permanent shift, from the transmitter to the receiver in the buffering of some information. The following paragraphs describe the operation of the receiver after positive NSP segment acknowledgment has occurred. The assumption is made that negative NSP segment acknowledgment will be performed by the receiver when appropriate.

The receiver must be willing to take any interrupt messages sent by the transmitter (i.e., provided the transmitter operates as indicated in Section 6.3.3). The receiver requests interrupt messages from the transmitter by sending an incremental request count to the transmitter in a Link Service Message. The transmitter takes the incremental request count and adds it to the current request count. The transmitter's request count must never exceed 127.

The receiver may stop the flow of normal data segments from the transmitter by sending a Link Service Message which sets the transmitter's data flow control switch to the "closed" value. The receiver can discard any received segments after this by negatively acknowledging any NSP segments which arrive after the flow control switch is closed.

Similarly the receiver may allow the flow of data messages from the transmitter by sending a Link Service Message which sets the transmitter's data flow control switch to the "open" value. This operation may be combined with the removal of the negative acknowledgment being given all NSP segments (if in effect).

The receiver can control data flow by a request count scheme similar to the one that exists for interrupt messages. Either dialogue messages or NSP segments containing dialogue data may be requested by this scheme.

6.3.2.1 Dialogue Message Request Count - The dialogue message request count scheme is employed when the receiver supports the message interface. With this approach, the receiving dialogue process allocates a buffer for each dialogue message it wants to receive. The receiver increments the message request count by one for each buffer that the receiving dialogue process supplies by sending a Link Service Message to the transmitter. The receiver may also accumulate receive requests and then send a Link Service Message requesting several data messages from the transmitter. Buffering spaces are provided for the first part of every dialogue message sent. Segments received (at the end of a dialogue message) for which there is no buffering space are positively acknowledged by the receiver and then discarded. If a segment is received for which there is room for only a portion of it, then the portion is placed in the users buffer and the remainder of the segment is discarded. When the last segment of such a dialogue message is discarded, the receiving dialogue process is informed that the receive is complete but that data was lost.

6.3.2.2 Segment Request Count - The segment request count scheme is employed when the receiver supports the segment interface. With this approach, the receiving dialogue process puts up a collection of buffers for receiving one or more dialogue messages.

The receiver calculates the number of segments that can be placed into each buffer, and increments the segment request count by that number by sending a Link Service Message to the transmitter. The receiver may also accumulate receive requests and then send a Link Service Message requesting several data segments from the transmitter. When a buffer is filled with the maximum number of segments it can hold, or when the last segment in a dialogue message has been placed in buffer, the buffer will be returned to the receiving dialogue process.

In the latter case, to avoid having the receiver return to the receiving dialogue process buffering space for segments that were requested but which are no longer available, a means of notification (that the receiver has buffering space for fewer segments than were previously requested) is employed. The receiver calculates the number of segments that it should have taken to completely fill the remaining space in the buffer. The negative of this number is sent back to the transmitter in the form of a Link Service Message which has the effect of decrementing the segment request count by the number specified.

Any NSP segments received following this action, for which there is no buffering space, may be negatively acknowledged.

6.3.3 Transmitter Operation -

6.3.3.1 Interrupt Messages - When the transmitter has an Interrupt Message to send, it examines the interrupt request count for the logical link. If the request count is not zero, it decrements the request count by one and sends the Interrupt Message. If the request count is zero, the Interrupt Message may not be sent. The transmitter operation at this point is implementation-dependent (i.e., it may queue the request internally, or it may inform the transmitting dialogue process that the Interrupt Message could not be sent).

When the transmitter receives a link services message containing an Interrupt Message request count, it normally adds the count from the message to the interrupt request count parameter for the logical link. If the result is greater than 127, or if the count from the message is negative, a link error event has occurred.

6.3.3.2 Data Segment - When the transmitter has a data segment to send, it examines the data flow control switch. If the switch has the "closed" value, the data segment may not be sent. The transmitter operation at this point is implementation-dependent. The transmitter may choose to queue the segment and act on a future change in value of the switch, or, it may inform the transmitting dialogue process that the data segment could not be sent.

If the data flow control switch has the "open" value, the data request switch is examined. If the data request switch has the value "none", the data segment may be transmitted. If it has any other value, the data request count parameter is examined.

If the data request count parameter is greater than zero a data segment may be sent. If the data request switch has the value "segment" and the data request count parameter is greater than zero, then the data request count will be decremented after the data segment is sent. If the data request switch has the value "message" and the data request count parameter is greater than zero, the data request count will be decremen4ed only if the segment is the last one in a dialogue message.

When the transmitter receives a Link Service Message containing a data request count, it normally adds the count to the data request count parameter for the logical link. If the data request count from the message is greater than 127 or less than -127, or, is negative and the data request scheme switch has the value "message", then a link error event has occurred. If the data request switch has the value "segment" the result might be that the data request count parameter would have a negative value.

When the transmitter receives a negative acknowledgment of an NSP segment containing data it will not automatically retransmit the NSP segment. Instead, it will re-evaluate the flow control parameters after performing the following computations. If the data request switch has the "segment" value, the data request count parameter is incremented by one for each NSP segment which has been negatively acknowledged. If the data request switch has the "message" value, the data message request count parameter is incremented by one for each NSP segment that was negatively acknowledged and that contained the last segment of a dialogue message. The receipt of one message containing an ACKNUM field may negatively acknowledge several segments implicitly (refer to Section 5.5.4). The processing described in this paragraph applies to each such segment.

6.4 Adjacent Node Initialization

Adjacent Node Initialization is a start-up procedure executed between two nodes that are directly connected on a physical link. Either node may choose to become active at any given time (even after a previously unsuccessful node initialization attempt).

The procedure consists of exchanging one or two messages containing information such as node name, node number, desired services, available services, and communication package version numbers. The procedure also includes the optional exchange of passwords as a verification feature.

Each node executing the procedure will determine if it is compatible with the other node. If each node decides that it is compatible with the other, then the physical link between them may be used for NSP messages. If either node decides that it is incompatible with the other, then it will not allow the physical link to carry NSP messages.

The adjacent node initialization procedure performs the following functions:

1. Identifies the node entering the network;
2. Performs a node verification (if required);
3. Provides information on the node;
4. Requests functions to be performed by the adjacent node; and
5. Sets and verifies network parameters.

Sections 6.4.1 and 6.4.2 describe the physical link parameters that are utilized to control the physical link and the password parameters associated with each node. Sections 6.4.3 and 6.4.4 describe the physical link states and events. Section 6.4.5 presents the physical link state table.

6.4.1 Physical Link Parameters - There are four parameters associated with each physical link in a node: a state control parameter, a read BLKSIZE parameter, a write BLKSIZE parameter, and a verification parameter. The state control parameter may take on one of three values: "off-line", "on-line", or "loop-back". The read BLKSIZE parameter contains the value that is placed in the field of the same name in any Node Initialization Message that is transmitted on the associated physical link. The write BLKSIZE parameter contains the value from the field of the same name from the last Node Initialization Message that has been received on the associated physical link. The verification parameter can take on one of two logical values: "verification required", or "verification not required". The value of the parameter has an effect on the operation of adjacent node initialization.

The way in which the state control, read BLKSIZE, and verification physical link parameters take on a value is beyond the scope of this specification. In some nodes, they may be under operator control; in others they may be defined when the node is generated. In any case, NSP operates under the following assumptions:

1. NSP can determine the value of the physical link parameters;

2. NSP cannot directly change the value of the parameters; and

3. If there are multiple physical links at a node, no more than one of them will have its associated state control link parameter set to the "loop-back" value.

If there are several physical links residing at a node and if each has its state control link parameter set to the "loop-back" value, NSP will respond correctly to the first link. NSP will treat the other links as if their state control link parameters were set to the "off-line" value.

6.4.2 Password Parameters - Two password parameters are associated with the identity of each node. The values of these passwords may be sent (Transmit Password) and received (Receive Password) during adjacent node initialization.

In addition, an NSPSIZE parameter is assigned to each node. The NSPSIZE parameter contains the value that is placed in the NSPSIZE field of a Node Initialization Message.

The password parameters are handled by NSP in a manner similar to that for the physical link parameters. That is, NSP can determine but not change the value of a parameter.

6.4.3 Physical Link States - The states of a physical link are:

1. OFF - the physical link is not being used by NSP, and the link parameter has the "off-line" value.

2. STARTING - the link parameter has the "on-line" or "loop-back" value, and NSP has issued a Start Command to the physical link control level. If the physical link supports dial-in, it is capable of answering a call in this state.

3. INITIALIZE - NSP has received a "start complete" from the physical link, has sent a Node Initialization Message, and is waiting for a Node Initialization Message to be received.

4. VERIFY - NSP has received a valid Node Initialization Message and is waiting for a Node Verification Message.

5. ON - NSP has received a valid Node Initialization Message and, if required, a valid Node Verification Message; the physical link may now be used by NSP on behalf of dialogue processes.

6.4.4 Physical Link Events - The events that can cause a transition from one state to another are:

1. a change in the value of the state control parameter;

2. a "Start Received", "Persistent Error", or "Maintenance Mode Entered" notification from the physical link control level;

3. a "Start Complete" notification from the physical link control level;

4. an NSP timeout threshold;

5. a received Node Initialization Message that is valid;

6. a received Node Verification Message that is valid;

7. a received message that is unexpected or invalid.

6.4.5 Physical Link State Table - A brief description of the physical link state transitions is provided in Table 6-1. In addition, notes are provided as an aid to clarify the operation of NSP with regard to adjacent node initialization.

Table 6-1. The Physical Link State Table

State	Event(s)	New State
OFF	State Control Link Parameter set to "on-line" or "loop-back"	STARTING
STARTING	State Control Link Parameter set to "off-line"	OFF
	"Start Complete" notification	INITIALIZE
INITIALIZE	State Control Link Parameter set to "off-line"	OFF
	State Control Link Parameter set to "loop-back"	STARTING
	State Control Link Parameter set to "on-line"	STARTING
	"Start Received" notification	STARTING
	NSP timeout threshold (see notes)	STARTING
	invalid Node Initialization or unexpected message received	STARTING
	valid Node Initialization received (verification required by this node)	VERIFY
	valid Node Initialization received (verification not required by this node)	ON
VERIFY	State Control Link Parameter set to "off-line"	OFF
	State Control Link Parameter set to "loop-back"	STARTING
	State Control Link Parameter set to "on-line"	STARTING
	"Start Received" notification	STARTING
	NSP timeout threshold (see notes)	STARTING
	invalid Node Verification or unexpected message received	STARTING
	valid Node Verification received	ON

Table 6-1. (Cont.) The Physical Link State Table

State	Event(s)	New State
ON	State Control Link Parameter set to "off-line"	OFF
	State Control Link Parameter set to "loop-back"	STARTING
	State Control Link Parameter set to "on-line"	STARTING
	"Start Received" notification	STARTING
	Node Initialization or Node Verification received	STARTING

Notes:

1. On entry to the STARTING state, a Start Command must be issued to the physical link control level. This command may be preceded by a Disconnect Link Command if the STARTING state is entered from any state other than OFF.

2. On entry to the INITIALIZE state, a Node Initialization Message must be sent.

3. Any event in the state table described as setting the values of the state control link parameter is an event in which the old value was different from the new value. Setting the parameter to its current value causes no state change.

4. This state table is presented entirely from the view of a single node. The decision to make the transition from INITIALIZE to either VERIFY or ON is dependent upon the Verification Parameter for the physical link over which adjacent node initialization is taking place. If the parameter has the "verification required" value then the node sets the "verification required" bit in the REQUEST field of the Node Initialization Message which is sent. A transition is made to the VERIFY state when a valid Node Initialization Message is received. If the Verification Parameter for the physical link has the "verification not required" value, then the "verification required" bit is not set in the REQUESTS field of the Node Initialization Message which is sent. A transition is made to the ON state when a Valid Initialization Message is received.

5. A timer may be started whenever the INITIALIZE or the VERIFY state is entered. If such a timer is used, an NSP timeout threshold event occurs when it expires. The timer is stopped on any transition from either state.

6. Whenever a valid Node Initialization Message is received with the "verification required" bit set in the REQUESTS field, a Node Verification Message must be the next message sent. The value for the PASSWORD field is the value of the transmit password parameter for the adjacent node. This operation has no effect on state transitions.

7. In the INITIALIZE state, an unexpected message is any message other than a Node Initialization Message.

8. In the VERIFY state, an unexpected message is any message other than Node Verification Message.

9. The Node Initialization and Node Verification messages must be sent without a RTHDR.

10. A Node Initialization Message is valid if:

    (a) The NODEADDR field contains either the number of the receiving node or of an adjacent node not yet connected via another physical link. In the former case, the state control link parameter must have the "loop-back" value. In the latter case, it must have the "on-line" value.
    (b) The NODENAME field contains either the name of the receiving node or of an adjacent node not yet connected via another physical link. In the former case, the state control link parameter must have the "loop-back" value. In the latter case, it must have the "on-line" value.
    (c) The NODENAME field contains only ASCII digits or upper case alphabetic characters. The number of digits or characters employed is at least 1 and no more than 6.
    (d) The NODEADDR field is not larger than 2 bytes. The value is greater than 1 and less than 241.
    (e) The value of the NSPSIZE field is less than or equal to the value of the BLKSIZE field.
    (f) The NSPSIZE field is valid according to the conditions described in note 14 below.
    (g) The receiving node can process messages from an NSP with the routing and communications versions indicated in the ROUTVER and COMMVER fields.
    (h) The FUNCTIONS field indicates that the node sending the Node Initialization messages can perform the functions required by the node receiving the message.
    (i) The INT subfield of the FUNCTIONS field contains either 0 or 7.
    (j) All fields of the Node Initialization Message are received.

11. A Node Verification Message is valid if the value of the PASSWORD field is present and is equal to the value of the Receive Password Parameter for the node sending the Node Verification Message.

12. It is valid for a node to request functions, via the REQUESTS field in a transmitted Node Initialization Message, from an adjacent node that are desired but not required.

13. A Phase II non-intercept implementation always sets the RINT subfield of the REQUESTS field in a transmitted Node Initialization Message to 3 and the INT subfield of the FUNCTIONS field to 0. A Phase II intercept implementation always sets the RINT subfield of the REQUESTS field in a transmitted Node Initialization Message to 0 and the INT subfield the FUNCTIONS field to 7.

14. The following conditions must exist for the NSPSIZE value to be valid:

    (a) It must be less than or equal to the NSPSIZE parameter for this node if this node is an intercept node, and
    (b) It must be greater than or equal to the NSPSIZE parameter for this node if the adjacent node is an intercept node.

15. The value of the BLKSIZE field from a received Node Initialization Message is always placed in the write BLKSIZE parameter for the physical link on which the Node Initialization Message was received.

6.5 Node/Link Failures

To effectively manage network activities, NSP must be capable of detecting, reporting, and maintaining a record of all node and link failures. When an error occurs, or a node is shutdown, NSP will notify the appropriate party(s) of this condition. Appendix D provides a list of the error codes sent by NSP.

6.5.1 Link Error Events - These are events which are detected by NSP as illegal operations performed by the NSP implementation which is managing the other end of a logical link and result in improperly formed data in certain NSP messages.

6.5.2 "No Path" Events - A "No Path" event occurs for a logical link when it is impossible to communicate with the remote node that is managing the remote end of the logical link.

A "No Path" event occurs under the following conditions:

1. Whenever a physical link leaves the "ON" state, a no path event occurs for each logical link associated with the physical link.

2. A Phase II implementation that is using the intercept function in an adjacent node may be informed by the intercept function (via a Disconnect Initiate or Disconnect Confirm Message) of the occurrence of a "No Path" event on a logical link. In this case, the "No Path" event and the event relating to the reception of the disconnect message cause the same state transitions.

APPENDIX A

GLOSSARY

Adjacent Nodes	Two nodes that are directly connected by a physical link regardless of that link's operational state. See Logically Adjacent Nodes.
Adjacent Routing	The ability of a node to route by name to a logically adjacent node.
Dialogue Level	The architectural level above NSP (defined by DNA).
Dialogue Message	A unit of information of variable length that may be sent by one dialogue process over a logical link to another dialogue process.
Dialogue Process	An entity residing above NSP in the DIGITAL Network Architecture that directly employs NSP's services.
Dialogue Segment	Part of a dialogue message.
Flow Control	The mechanism by which NSP segments containing data from a dialogue process are allowed to be sent (or prevented from being sent) from one NSP implementation to another for the purpose of buffer management.
Guaranteed Delivery	The guarantee by a data delivery mechanism to a source dialogue process that a unit of information sent by the source dialogue process will be delivered to an area of storage (e.g., a buffer) accessible to the destination dialogue process. If delivery is not possible the source dialogue process should be informed of the failure.
Guaranteed Sequentiality	The guarantee by a data delivery mechanism to a source dialogue process that data sent in a particular order by the source dialogue process will be delivered in the same order to the destination dialogue process.

Intercept	A function that may exist in a node that is adjacent to a Phase II implementation of NSP. The intercept function allows a Phase II implementation to communicate correctly with non-adjacent nodes.
Logical Link	A logical, full duplex communication path, maintained by NSP between two dialogue processes.
Local Link Address	The logical link address by which an NSP implementation identifies a particular logical link.
Logically Adjacent Nodes	Physically adjacent nodes in which both perceive the state of the physical link to be ON. Both nodes will have the value of the state control link parameter for the physical link set to "on-line".
Node	An implementation of NSP.
NSP Message	A unit of information sent by one implementation of NSP to another.
NSP Segment	A message or unit of information that is assigned a number by NSP.
Physical Link Control Level	The architectural level below NSP (defined by DNA).
Receiver	The NSP implementation that receives the data on the logical link. This term is used for explanatory purposes only, since data transmission on a logical link is possible in either direction.
Routing by Name	The ability of a node to map a node name into one of three logical values: "self", "not-self but accessible" or "not accessible", and if the value is "not self but accessible", to map the node name to a particular physical link.
Transmitter	The NSP implementation that transmits the data on the logical link. This term is used for explanatory purposes only, since data transmission on a logical link is possible in either direction.

APPENDIX B

LOGICAL LINK STATE TABLES

Tables B-1 through B-8 are the Logical Link State Tables.

The following notes apply to all tables:

1. N/C means no change in state.

2. N/A means not applicable. N/A represents either an error on the part of the dialogue process or it is an impossible event.

3. A plus sign (+) indicates an event that has a high probability of occurring but which should be ignored.

4. A minus sign (-) indicates an event that has a low probability of occurring or is a protocol error. This event should be ignored.

5. A zero (0) indicates an event whose occurrence has no effect on the state of a logical link and which is either described under logical link management or is handled in an implementation dependent manner.

6. If a DATA message containing an ACKNUM field is received, the ACKNUM field is processed first as if it had been received in a separate ACK message. The rest of the DATA message is then processed.

7. If a Disconnect Confirm message is received without a REASON field, it is processed as if it contained an "unspecified error condition" code in a REASON field.

Table B-1. Logical Link Table for the IDLE State

Event	Response Message	New State	Notes
Dialogue process requests connection	CI	CIS N/C	1 2
Dialogue process accepts connection	N/A	N/C	
Dialogue process rejects connection	N/A	N/C	
Dialogue process disconnects or requests link abort	N/A	N/C	
Dialogue process aborts	(0)	N/C	
A Connect Initiate Message is received	 DC	CIR N/C	3 4
A Connect Confirm Message is received	DC	N/C	5
An ACK message is received	DC	N/C	5
A NAK message is received	DC	N/C	5
A DATA, Interrupt, or Link Service Message is received	DC	N/C	5
A Disconnect Initiate Message is received	DC	N/C	5
A Disconnect Confirm Message is received		N/C	
A "no path" event occurs	N/A	N/C	
A link error event occurs		N/C	6

Notes:

1. The contents of DSTADDR must be 0, and the contents of SRCADDR must be the local link address. The value of SEGSIZE is equal to the minimum of:

   a. The value of the NSPSIZE parameter for this node minus the maximum length of an NSP header for a data message, or
   b. the value of SEGSIZE from the dialogue process if the segment interface is used.

2. If the destination node is not accessible, the dialogue process is informed that the connection cannot be made.

3. This operation applies to a Phase II implementation which received a Connect Initiate Message with acceptable SERVICES, process identification information, access control information and no reserved bits in the MENU field set. The SERVICES field is acceptable, if: a) it requests no services that are unavailable, b) it contains the defined value in bit positions defined with constant values, and c) if it contains no extensions defined as reserved. See Section 6.2 for a discussion of acceptable process identification and access control information. The INFO field is always acceptable. Any undefined information contained in the field or any extensions to its length are ignored. If the interface is a segment interface, the value of SEGSIZE given to the dialogue process is the minimum of:

   a. the contents of the SEGSIZE field from the Connect Initiate Message, or
   b. the value of the NSPSIZE parameter for this node minus the maximum length of an NSP header for a data message.

4. This operation applies when the received Connect Initiate Message contains unacceptable information. The value of the DSTADDR and SRCADDR fields from the received Connect Initiate Message are reversed in order to form the DC message.

5. The values of the DSTADDR and SRCADDR fields are reversed to form the Disconnect Confirm Message as in Note 4.

6. A link error event is the receipt of any message with a zero SRCADDR field or a Connect Initiate Message which is too short.

Table B-2. Logical Link Table for the CIR State

Event	Response Message	New State	Notes
Dialogue process requests connection	N/A	N/C	
Dialogue process accepts connection	CC	RUN	1
Dialogue process rejects connection	DI	DIS	
Dialogue process disconnects or requests link abort	N/A	N/C	
Dialogue process aborts	DI	DIS	
A Connect Initiate Message is received	(-)	N/C	
A Connect Confirm Message is received	(-)	N/C	
An ACK message is received	(-)	N/C	
A NAK message is received	(-)	N/C	
A DATA, Interrupt, or Link Service Message is received	(-)	N/C	
A Disconnect Initiate Message is received	(-)	N/C	
A Disconnect Confirm Message is received	(-)	N/C	
A "no path" event occurs		IDLE	2
A link error event occurs	(-)	N/C	

Notes:

1. The value of SEGSIZE in the Connect Confirm Message is equal to the minimum of:

   a. the value of the NSPSIZE parameter for this node minus the maximum length of an NSP header for a DATA message, or

   b. the value of SEGSIZE form the dialogue process if the segment interface is used.

2. The dialogue process that received the connect request should be informed of the event.

Table B-3. Logical Link Table for the CIS State

Event	Response Message	New State	Notes
Dialogue process requests connection	N/A	N/C	
Dialogue process accepts connection	N/A	N/C	
Dialogue process rejects connection	N/A	N/C	
Dialogue process disconnects or requests link abort		IDLE	
Dialogue process aborts		IDLE	
A Connect Initiate Message is received	(-)	N/C	
A Connect Confirm Message is received	 DC 	RUN IDLE IDLE	1 2 3
An ACK message is received		N/C IDLE	4 5
A NAK message is received	(-)	N/C	
A DATA, Interrupt, or Link Service Message is received	(+)	N/C	
A Disconnect Initiate Message is received	DC	IDLE IDLE	6 7
A Disconnect Confirm Message is received		IDLE	8
A "no path" event occurs		IDLE	9
A link error event occurs	(-)	N/C	

Notes:

1. This operation applies to a Phase II implementation that has received a Connect Confirm Message with an acceptable SERVICES field. A SERVICES field is acceptable if it requests no unavailable services and if it contains no extensions defined as reserved. The INFO field is always acceptable. Any undefined information contained in the field or any extension to its length are ignored. If the interface is a segment interface, the value of SEGSIZE given to the dialogue process (as an output parameter from the connect request) is equal to the minimum of:

   a. the contents of the SEGSIZE field from the message, or
   b. the value of the NSPSIZE parameter for this node minus the maximum length of an NSP header for a data message.

   Note that the data base for this link had no previous remote link address defined for it, so the SRCADDR field is ignored when testing for a logical link match (provided that it is non-zero).

2. This operation applies to the action of NSP when the received Connect Confirm Message contains an unacceptable SERVICES field. The contents of the DSTADDR and SRCADDR fields are swapped in order to form the Disconnect Confirm Message.

3. This operation applies to the receipt of a Connect Confirm Message containing a zero SRCADDR field or a Connect Confirm Message that is too short (i.e., does not contain all the required fields or a field terminates prematurely).

4. This operation applies to a valid ACK message. An ACK message is valid if and only if: (a) the value of SRCADDR is 0, (b) the acknowledgment pertains to a data message with segment number 0, and (c) no reserved bits are set in the ACKNUM field.

5. This operation applies to receiving an invalid Acknowledgment Message. See Note 4 for a definition of valid.

6. The contents of the SRCADDR field are ignored, provided that they are non-zero, and the DSTADDR and SRCADDR fields are swapped in order to form the Disconnect Confirm Message. Note 9 applies.

7. This processing applies to a received Disconnect Initiate Message containing a zero SRCADDR field or to a received Disconnect Initiate Message which is too short.

8. The value of SRCADDR must be 0 to match an existing logical link in the CIS state (in this case note 9 applies); otherwise, the Disconnect Confirm Message applies to an IDLE logical link.

9. The dialogue process should be informed that the connection request failed.

Table B-4. Logical Link Table for the RUN State

Event	Response Message	New State	Notes
Dialogue process requests connection	N/A	N/C	
Dialogue process accepts connection	N/A	N/C	
Dialogue process rejects connection	N/A	N/C	
Dialogue process disconnects or requests link abort	DI	DIS	
Dialogue process aborts	DI	DIS	
A Connect Initiate Message is received	(-)	N/C	
A Connect Confirm Message is received		N/C	1
AN ACK message is received	(0)	N/C	
A NAK message is received	(0)	N/C	
A DATA, Interrupt, or Link Service Message is received	(0)	N/C	
A Disconnect Initiate Message is received	DC	IDLE	2
A Disconnect Confirm Message is received		IDLE	2
A "no path" event occurs		IDLE	2
A link error event occurs	DC	IDLE	3

Notes:

1. This operation occurs when a duplicate Connect Confirm Message is received. A duplicate is ascertained by examination of the Connect Confirm Message only through the SRCADDR field. The Connect Confirm Message is ignored.

2. Inform the dialogue process.

3. A link error event is any one of the following:

   a. Receipt of an ACK, NAK, DATA, Interrupt, or Link Service Message that has reserved bits set in the ACKNUM field.

   b. Receipt of a data message that is longer than the SEGSIZE value for receiving on the logical link.

   c. Receipt of a data, Interrupt, Link Service, Acknowledgment, or Disconnect Initiate Message that is too short or is missing a field.

   d. Receipt of a Link Service Message that has the LSFLAGS extended or reserved bits set.

   e. Receipt of a Link Service Message that contains a non-zero FCVAL field and that matches a logical link on which there is no request count flow control at the remote end.

Table B-5. Logical Link Table for the DIS State

Event	Response Message	New State	Notes
Dialogue process requests connection	N/A	N/C	
Dialogue process accepts connection	N/A	N/C	
Dialogue process rejects connection	N/A	N/C	
Dialogue process disconnects or requests link abort	(0)	N/C	1
Dialogue process aborts	(+)	N/C	
A Connect Initiate Message is received	(-)	N/C	
A Connect Confirm Message is received	(-)	N/C	
An ACK message is received	(+)	N/C	
A NAK message is received	(+)	N/C	
A DATA, Interrupt, or Link Service Message is received	(+)	N/C	
A Disconnect Initiate Message is received	DC	IDLE IDLE	2 3
A Disconnect Confirm Message is received		IDLE	
A "no path" event occurs		IDLE	
A link error event occurs	(-)	N/C	

Notes:

1. The possibility of the occurrence of this event is implementation-dependent.

2. This operation applies to the receipt of a Valid Disconnect Initiate Message.

3. This operation applies to the receipt of a Disconnect Initiate Message that is too short.

APPENDIX C

OBJECT TYPES

The object type code values that have been defined are listed below, expressed as octal byte values.

DIGITAL reserves the right to add object types and to make changes to the descriptor formats used by the object types.

Object Type	Descriptor Format	Process Type
000	1 or 2	General task, User process
001	0	File Access (FAL/DAP-Version 1)
002	0	Unit Record Services (URDS)
003	0	Application Terminal Services (ATS)
004	0	Command Terminal Services (CTS)
005	0	RSX-11M Task Control-Version 1
006	0	Operator Services Interface
007	0	Node Resource Manager
010	0	IBM 3270-BSC Gateway
011	0	IBM 2780-BSC Gateway
012	0	IBM 3790-SDLC Gateway
013	1 or 2	TPS Application
014	1 or 2	RT-11 DIBOL Application
015	0	TOPS-20 Terminal Handler
016	0	TOPS-20 Remote Spooler
017	0	RSX-11M Task Control-Version 2

(continued on next page)

Object Type	Descriptor Format	Process Type
020	0	TLK Utility
021	0	File Access (FAL/DAP-Version 4)
022	0	RSX-11S Remote Task Loader
023	0	NICE Process
024 - 076	0, 1, 2	Reserved for DECnet Use
077	0	DECnet RSX Test Tool
100 - 177	0, 1, or 2	Reserved for DECnet control
200 - 377	0, 1, or 2	Reserved for customer extensions

APPENDIX D

DISCONNECT ERROR CODES

The following error code values have been defined for use in the REASON field of either the Disconnect Initiate or Disconnect Confirm NSP messages:

Error Code	Meaning
0	No error.
1	Resource allocation failure.
2	Destination node does not exist.
3	Node shutting down (for use by nodes not wishing to accept new links).
4	Destination process does not exist.
5	Invalid process name field.
6	Destination process queue overflow.
7	Unspecified error condition.
8	Third party aborted the logical link.
9	Link abort by dialogue process.
24	Flow control violation-illegal FCVAL in Link Services Message.
32	Too many connections to node.
33	Too many connections to destination process.
34	Access not permitted-Unacceptable RQSTRID or PASSWORD.
35	Logical link SERVICES mismatch.
36	Unacceptable ACCOUNT information-unauthorized or account balance unacceptable.
37	SEGSIZE too small.
38	Dialogue process aborted, timed out, or cancelled request.
39	No path to destination node.
40	Flow Control Failure.
41	DSTADDR logical link does not exist.
42	Confirmation of Disconnect Initiate.
43	Image data field too long-RQSTRID, PASSWORD, ACCOUNT, USRDATA (in Connect Initiate and Connect Confirm Messages), and DATA (in Disconnect Initiate Messages).

The following list indicates which error codes may occur in specific situations.

Situation	Error Codes
Connect Initiate rejected	1, 2, 3, 4, 5, 6, 32, 33, 34, 36, 38, 39, 43
Connect Confirm rejected	35, 37, 38, 41, 43
Disconnect sent	1, 2, 3, 4, 5, 6, 8, 24, 32, 33, 34, 36 37, 38, 39, 40, 41, 42, 43

Notes:

1. Error Code 0 should be used for Disconnect Initiate Messages that result from a synchronous disconnect request from the dialogue process. Error Code 9 should be used for Disconnect Initiate Messages that result from a link abort request from the dialogue process.

2. Error 8 should be used when an operator or some system process has the capability to disconnect logical links.

3. Error 24 should be used for the following error conditions:

    a. The FCVAL field received in a Link Service Message for a data subchannel that is segment flow controlled would result in a cumulative count of segment requests greater than +127 or less than -127.

    b. The FCVAL field received in a Link Service Message for a data subchannel that is message flow controlled was negative, or would result in a cumulative count of message requests greater than +127.

    c. The FCVAL field received in a Link Service Message for a data subchannel that is not flow controlled is non-zero.

    d. The FCVAL field received in a Link Service Message for the Interrupt/Link Services subchannel was negative, or would result in a cumulative count of interrupt message requests greater than +127.

4. Error 43 should be used for illegal image fields, not error 34.

5. Examples of the use of Error 36 are:

    a. Account not authorized for specified RQSTRID.

    b. Balance of the account insufficient to allow new connection.

    This error should not be used for an illegal image account field. This is covered by Error 43.

6. Error code 40 should be used for the following error conditions:

   a. Data message received on a message flow controlled link when the request count is zero.

   b. Interrupt Message received when the interrupt request count is zero.

   Note that data may be received on a segment flow controlled link when the request count is zero or negative.

7. Error code 43 applies to process descriptors, RQSTRID, PASSWORD, ACCOUNT, and USRDATA in Connect Initiate Messages and the DATA field in a Disconnect Initiate Message. It does not apply to image fields in route headers or in Node Initialization Messages.

APPENDIX E

TASK-TO-TASK INTERFACE REQUIREMENTS

E1.0 INTRODUCTION

This appendix describes the task-to-task interface common to all operating systems that implement DECnet. The interface serves to make a subset of the facilities provided by the logical link control layer available to users in a consistent fashion. Any system claiming to support DECnet task-to-task operation must provide the functions presented below.

E2.0 GOALS

In defining a common subset interface, the following goals should be met:

1. Provide a common subset of task-to-task functions across all operating systems.

2. Ensure that users of the task-to-task interface can write applications that can communicate in a heterogeneous network.

3. Guarantee that users of the task-to-task interface do not jeopardize the integrity of the network.

4. Allow for the upward compatible extension of the interface in future releases of the operating systems and DECnet.

E3.0 FUNCTIONS AND PARAMETERS

This following section lists the functions and parameters that must be made available to the dialogue process. Input parameters describe information given to NSP by the dialogue process; output-parameters describe information given to the dialogue process by NSP.

E3.1 Connect Request

Input parameters:

Destination Node Name
Destination Process Identifier
Access Control Data
User Data Sent

Output parameters:

Connect Accept, Connect Reject, Rejection by destination node indicator
User Data Received

The destination node name is a printable ASCII character string that specifies the target node for the logical link. The interface must provide for a string length of six characters. The printable ASCII characters in this string are limited to numeric and upper case alphabetic characters.

If the destination node name is less than six characters in length, the way in which significant characters are defined in this field is implementation-dependent.

The destination process identifier addresses a dialogue process in the target node to which the request for a logical link should be forwarded. This parameter consists of three subfields:

Format Identifier
Object Type Number
Object Descriptor

Collectively, these fields address either a unique dialogue process or a generic capability in the target node. See Section 2.4.2 for further definition of these fields. The interface must provide for a format identifier, an object type number, and up to 16 bytes of 7-bit ASCII. The interface must also provide a means whereby the length of the ASCII data can be supplied.

The access control data is used by the target node to determine whether the source process has sufficient privilege to access the destination process. The access control data consists of these subfields.

Requestor ID
Password
Accounting Information

In systems where access control is not provided below the user level, the interface must provide for up to 16 bytes of Requestor ID, an 8-byte binary password, and up to 16 bytes of Accounting Information string. The interface must also provide a means whereby the length of the Requestor ID and the Accounting Information can be supplied. In systems that provide access control below the user level, access control information need not cross the subset interface. User data may be up to 16 bytes in length.

The Connect Request operation can complete in at least three ways; a rejection by the target node, a rejection by the addressed process, or an acceptance by the addressed process. The interface must guarantee that these three conditions are distinguishable, and, when the connection is rejected or accepted by the destination process, any data provided by the Connect Accept or Connect Reject (up to 16 bytes) is returned to the source process.

E3.2 Receive Connect Request

Input parameters:

buffer to receive output parameters

Output parameters:

Source node name
Source process identifier
Access control data
User data

The parameters returned to the dialogue process have the same interpretation as do the corresponding destination parameters for Connect Request.

Access control data may not be returned to the dialogue process in systems that provide access control validation below the user level.

It is not mandatory that a destination process issue a "Receive Connect Request" prior to the sending of a Connect Request by the source process. The purpose of defining a Receive Connect Request function is to guarantee that the destination process is provided a mechanism to retrieve information about and from the calling process. It may be a requirement in some systems that a network set-up command be issued to provide the process name of the receiving process.

E3.3 Connect Accept

Input parameters:

User data

User data may be up to 16 bytes in length. The decision whether or not to pass the data must reside with the process that initiated the connect request.

E3.4 Connect Reject

Input parameters:

User data

The same considerations apply as for user data on Connect Accept.

E3.5 Disconnect (Synchronous with Data) and Abort

Input parameters:

User data

User data may be up to 16 bytes in length. If the disconnect operation completes normally, the data is delivered to the target dialogue process; otherwise, the process initiating the disconnect is notified of the abnormal condition. The target process must be informed of the type of disconnect:

a. by partner's disconnect

b. by partner's abort

c. by NSP

There is no way to guarantee that the target process will correctly process the data. We do however, want to guarantee that if the user detects the disconnect, that user will receive any data sent with that disconnect.

The two processes that have a link between them must cooperate in order to use the synchronous disconnect without a deadlock. Specifically, the synchronous disconnect request should normally be made by only one of the processes. If each process has sent data and then requests a synchronous disconnect before the data is acknowledged, then the link will never disconnect, because each end of the link will be negatively acknowledging received data while waiting for positive acknowledgment of transmitted data before allowing the disconnect to occur.

E3.6 Transmit Data

The parameters are as follows:

Dialogue process data
End of dialogue message indicator

E3.7 Receive Data

The Receive Data parameters are as follows:

Dialogue process data
End of dialogue message indicator

The subset interface provides the user with an interface for transferring dialogue messages to a remote user. Depending on implementation and operating system requirements, users associate one or more transmit data requests with a logical unit of information. This unit of information is presented to the target process so that it is meaningful in the context of the operating system at the target node. Thus, a transmitting process is assured that a dialogue message sent appears as a dialogue message received, regardless of the specific system implementation.

Two generic mechanisms for transmission and reception of data have been identified:

1. One dialogue message in one buffer.

2. One dialogue message spans multiple buffers.

In the first form of interface, end of message is implied for each request and therefore does not explicitly appear at the interface. On transmit, each request results in one or more NSP segments being transmitted, with the end of message indicator set in the last NSP segment. The source process is notified that the request is complete when the dialogue message is received by the destination NSP or that the source NSP has buffered the message and will ensure its delivery. On receive, segments are assembled into dialogue messages such that they always start at the beginning of a buffer. The destination process is notified only when the end of a dialogue message is received or when the buffer overflows. In buffer overruns, the excess data is lost, and the user must be notified of the condition.

In the second form of the interface, where one dialogue message spans multiple buffers, end of message is explicitly specified by the user. The size of each buffer is not specified, but may be restricted in some systems to the size of the block that can be sent on the physical link. Dialogue messages must always start on a buffer boundary.

There is no defined relation between user requests to the user interface and the use of flow control mechanisms by NSP. That is, there is no defined user request that is guaranteed to cause NSP to send a Link Services message.

E3.8 Transmit Interrupt Data

Input parameter: User Data

User data may be up to 16 bytes in length. If the interrupt data cannot be sent, the user must be notified.

E3.9 Receive Interrupt Data

Input parameters: Buffer to Store Interrupt Data

Output parameters: User Data

User data may be up to 16 bytes in length. Each operating system must provide a mechanism for passing unsolicited interrupt messages to users. Ideally, receipt of an interrupt message should interrupt the destination process, but this is not possible on all systems. If interrupt is impossible, the interrupt message should be placed ahead of all data in the target's receive queue.

APPENDIX F

PHASE II INTERCEPT OPERATION

F1.0 TOPOLOGICAL RESTRICTIONS

This operation pertains only to a Phase II intercept node. Such a node is topologically restricted to being the center of a "star" network configuration. The term "satellite" refers to any node in such a configuration other than the center of the star. All satellites in such a configuration must be Phase II implementations without intercept.

F2.0 PURPOSE OF THE INTERCEPT NODE

The primary purposes of a Phase II intercept node are to:

1. inform a source satellite if a destination satellite to which a Connect Initiate Message has been sent is inacessible;

2. forward NSP data messages to existing satellites; and

3. inform a given satellite if a second satellite with which the given satellite was communicating has become inaccessible.

F3.0 OPERATION OF A PHASE II INTERCEPT NODE

Intercept operation occurs when one satellite establishes a logical link (or attempts to establish a logical link) with another satellite. For each logical link, the intercept node must establish a data base containing: (a) the identity of each satellite; (b) the logical link address by which each satellite identifies the link; and (c) the state each satellite perceives the logical link to be in.

The operation of the intercept node is summarized by the following rules:

NOTE

> The operation described below applies to intercept operation only. The intercept node operates on a logical link between itself and any satellite as a normal Phase II implementation. In particular, note 6 is intended to describe the operation of an intercept node that receives a Data Message without a RTHOR

destined for a satellite. It is acceptable to receive a Data Message without a RTHDR destined for the intercept node itself. In this case, the operation of the intercept node is as described in the main body of this specification.

1. When the intercept node detects a protocol error on the part of a satellite, it reinitializes the physical link to the satellite. See note 2.

2. When a satellite becomes unreachable (either because of a failure or reinitialization of the physical link to it) all logical links that terminated in the satellite are examined.

   If the logical link is not in the CIR state from the point of view of the other (still reachable) satellite, then a Disconnect Initiate Message is sent to the other satellite.

   If the logical link is in the CIR state from the point of view of the other (still reachable) satellite, then the data base for the logical link is marked as "disconnect required".

   If the other satellite in which the logical link terminated is unreachable, the data base for the logical link is removed or initialized to an "available for new logical link" state.

3. When a Connect Initiate Message is received from a satellite, the DSTNODE field is examined. If the field contains the name of a reachable satellite, then the message is sent to that satellite. If the field contains a name that is not equal to the name of a reachable satellite, the intercept node returns an appropriate Disconnect Confirm Message to the sender with a REASON code indicating that the destination is inaccessible.

4. When a Connect Confirm Message is received from a satellite, the logical link to which the message applies is ascertained by examining the DSTNODE and DSTADDR fields. If this cannot be done, the sender has committed a protocol error. If it can be done, the logical link data base is examined. If the logical link is marked as "disconnect required", an appropriate Disconnect Confirm Message is returned to the sender with a REASON code indicating that the destination is inaccessible. Otherwise, the message if forwarded to the destination satellite.

5. When an NSP message other than a Data Message Interrupt Message, Link Service Message, or Acknowledgment Message is received from a satellite, the message is forwarded to the destination satellite if it is reachable; otherwise, the message is discarded. Notes 3 and 4 apply, however.

6. When a Data Message Interrupt Message, Link Service Message, or Acknowledgment Message is received from a satellite it is examined. If it has a RTHDR, it is handled as any other NSP message (see note 5). If the message has no RTHDR, the destination satellite, is ascertained by examining the SCRADDR field. If the destination satellite cannot be ascertained (because the SRCADDR field contained an unknown logical link address), then the sender has committed a protocol error; otherwise, the Data Message is forwarded, as is, to the appropriate destination satellite.

APPENDIX G

REVISION HISTORY

This appendix describes the general changes in NSP since the version (generally referred to as Version 1.0) that was specified in "Design Specification for Network Services Protocol", dated 10 July 1975.

Because there have been several major changes to NSP since Version 1, and because no previous versions have been standardized, the changes discussed in this appendix are the general, functional changes to NSP rather than specific, operational changes. In particular, no details of changes to message formats are described. The changes are described according to category (i.e., existing features revised, new features added, old features removed).

Existing Features Revised:

1. Process addressing has been enhanced from a form that was oriented toward RSX-11M to a more general form.

2. Operation has been described in terms of state tables.

3. Some message types were removed as redundant since the functions performed by the removed types were ascertained via the state tables to be performed by other message types. The message types removed were Connect Reject (by NSP), Disconnect Reject (by user), and Disconnect Abort.

4. The routing information that may be included with a message has been changed to allow routing by name only.

5. Flow control on a logical link was enhanced to add "on/off" control and to allow request counts to be applied to messages or segments or to not be applied at all; furthermore, when request counts are used, they are always incremental. In Version 1, request counts for a logical link were always required, applied only to data messages, and were incremental for "unnumbered" logical links and relative to acknowledgment numbers for "numbered" logical links.

6. Flow control information flows on a numbered data subchannel rather than via unnumbered control messages. As a result, the Request Link Status message type and the Confirm Request Count message have both been removed, and the Link Status message type has been replaced by a Link Service Message type.

7. The restriction of a maximum of 16 segments per message has been removed. There is no maximum currently.

8. Message blocking has been removed.

9. The disconnect function on a logical link became an abort function.

10. Interrupt message transfer has been put under flow control. In Version 1, there was no control over the flow of interrupt messages.

New Features Added:

1. Negative acknowledgment of data segments was added.

2. The synchronous disconnect function on a logical link was added.

3. Access control information has been added to the Connect Initiate Message.

4. A procedure for initializing two adjacent nodes was added.

5. The general intercept concept and the operation of Phase II intercept were added.

Old Features Removed:

1. Message tracing (which was only loosely defined in Version 1) was removed.

2. Adaptive routing (as a possible function of NSP) was removed. As a result, the routing path message type was removed.

3. The echo maintenance function was removed. As a result, the Echo message type and Echo Reply message type were removed.

4. The Error Message was removed.

5. The ability to determine remote node configuration was removed from NSP. As a result, the Request Configuration Message type and the Configuration Message type were removed.

digital equipment corporation

Printed in U.S.A.