H300IPC NetEx/IP®

for Unisys OS2200 Systems

Release 7.4.3

Software Reference Manual

REF-H300IPC-7.4.3

Revision Record

Revision	Description
7.1 (June 2011)	This first revision of the document reflects the first generally available release of H300IPC, release 7.1.8.
7.2.2 (September 2011)	This revision of the document reflects revisions related to the general release of H300IPC release 7.2. • A change was made to document revision tracking. • The description for the new LONGMSG = 1 parameter was added to the Initialization statements in this manual • The description for the new OPTION=LONGMSG parameter was added to the HOST statement parameters in the "Configuration Management" section. • The description for the new OPTION=ALTFIRST parameter was added to the HOST statement parameters in the "Configuration Management" section. • The description for the new initialization statement IPCKEY= was added to the "Initialization Statements" section. • The number of copies of Netex that may communicate with an IPC has been reduced from 8 to 6.
7.3 (November 2011)	• Formal release. Miscellaneous fixes documented in the MTU.
7.3.1 (January 2012)	• Formal release. Miscellaneous fixes documented in the MTU.
7.3.2 (February 2012)	• Formal release. Miscellaneous fixes documented in the MTU. • Corrected the allowed value range for the "NTXNUM" initialization parameter.
7.4 (January 2013)	• Added DNS lookup option for GNA-IP resolution
7.4.1 (July 2013)	• Miscellaneous fixes in the documentation.
7.4.2 (May 2014)	• Switch command added to netex and IPC • IPC KILL command added to IPC
7.4.3 (Nov 2014)	• Groupname or hostname in connect protocol

You may submit written comments to:

Network Executive Software, Inc.
Publications Department
6420 Sycamore Lane, Suite 300
Maple Grove, MN 55369
USA

Comments may also be submitted by e-mail to support@netex.com or, by visiting our web site http://www.netex.com.

Always include the complete title of the document with your comments.

Preface

This manual describes Network Executive Software's H300IPC NETwork EXecutive (NetEx) for the Unisys 2200 Series operating system. This manual corresponds to the version of NetEx referred to in the Revision Record.

This manual is divided into the following sections:

"Introduction" introduces NetEx and is intended for all readers.

"Understanding Intertask Communication", "Understanding the NetEx Request Block", "Assembler Interface", and "FORTRAN Interface", describe how to program using NetEx and is intended for the application programmer.

"Installation", "Initialization Statements", and "Configuration Management" describe NetEx installation and are intended for the systems programmer or whoever is installing NetEx.

"Operator Interface" describes the operator interface to NetEx.

The appendices include a list and description of the error codes and messages issued by NetEx.

Readers are not expected to be familiar with NetEx before using this manual. However, an understanding of programming and using the host operating system is required.

Reference Material

The following manuals contain related information.

Title and Description

"C" Configuration Manager and NETEX Alternate Path Retry (APR) User Guide

H301 Bulk File Transfer (BFX'") Utility for Unisys Software Reference Manual

Notice to the Reader

The material contained in this publication is for informational purposes only and is subject to change without notice. Network Executive Software is not responsible for the use of any product options or features not described in this publication, it and assumes no responsibility for any errors that may appear in this publication. Refer to the revision record (at the beginning of this document) to determine the revision level of this publication.

Network Executive Software does not by publication of the descriptions and technical documentation contained herein, grant a license to make, have made, use, sell, sublicense, or lease any equipment or programs designed or constructed in accordance with this information.

This document may contain references to the trademarks of the following corporations:

Corporation Trademarks and Products

Network Executive Software **NetEx, BFX, PFX, USER-Access**

Unisys Corporation **Unisys, 2200, ClearPath, Dorado, MASM, FTN**

These references are made for informational purposes only.

The diagnostic tools and programs described in this manual are **not** part of the products described.

Notice to the Customer

The installation information supplied in this document is intended for use by experienced System Programmers.

Software Modification Policy

Modifications to H300IPC that are not specifically authorized by a representative of Network Executive Software are prohibited.

Any unauthorized modifications to H300IPC may affect its operation and/or obstruct NetEx Software's ability to diagnose problems and provide corrections. Any work resulting from unauthorized modifications shall be paid by the customer at NetEx Software's then-current support rates and may result in the immediate termination of warranty/support coverage.

Document Conventions

The following notational conventions are used in this document.

Format	Description
`displayed information`	Information displayed on a CRT (or printed) is shown in `this font`.
`user entry`	*`This font`* is used to indicate the information to be entered by the user.
UPPERCASE	The exact form of a keyword that is not case-sensitive or is issued in uppercase.
MIXedcase	The exact form of a keyword that is not case-sensitive or is issued in uppercase, with the minimum spelling shown in uppercase.
bold	The exact form of a keyword that is case-sensitive and all or part of it must be issued in lowercase.
lowercase	A user-supplied name or string.
value	Underlined parameters or options are defaults.
<label>	The label of a key appearing on a keyboard. If "label" is in uppercase, it matches the label on the key (for example: <ENTER>). If "label" is in lowercase, it describes the label on the key (for example: <up-arrow>).
<key1><key2>	Two keys to be pressed simultaneously.
No delimiter	Required keyword/parameter.

Glossary

ASCII: Acronym for American National Standard Code for Information Interchange.

buffer: A contiguous block of memory allocated for temporary storage of information in performing I/O operations. Data is saved in a predetermined format. Data may be written into or read from the buffers.

Configuration Manager: A utility that parses a text NCT file into a PAM file.

header: A collection of control information transmitted at the beginning of a message, segment, datagram, packet, or block of data.

host: A data processing system that is connected to the network and with which devices on the network communicate. In the context of Internet Protocol (IP), a host is any addressable node on the network; an IP router has more than one host address.

Internet Protocol (IP): A protocol suite operating within the Internet as defined by the *Requests For Comment* (RFC). This may also refer to the network layer (level 3) of this protocol stack (the layer concerned with routing datagrams from network to network).

ISO: Acronym for International Standards Organization.

Network Configuration Table (NCT): An internal data structure that is used by the NetEx configuration manager program to store all the information describing the network.

NETwork EXecutive (NetEx): A family of software designed to enable two or more application programs on heterogeneous host systems to communicate. NetEx is tailored to each supported operating system, but can communicate with any other supported NetEx, regardless of operating system.

NetEx is a registered trademark of Network Executive Software.

Open Systems Interconnection (OSI): A seven-layer protocol stack defining a model for communications among components (computers, devices, people, and et cetera) of a distributed network. OSI was defined by the ISO.

processor interface (PI): A PI interfaces a minicomputer with an adapter. The PI is a board(s) that contains a microprocessor and memory. The processor interface is generally installed in the host. Some types of PIs contain NetEx.

path: A route that can reach a specific host or group of devices.

TCP/IP: An acronym for Transmission Control Protocol/Internet Protocol. These communication protocols provide the mechanism for inter-network communications, especially on the Internet. The protocols are hardware-independent. They are described and updated through *Requests For Comment* (RFC). IP corresponds to the OSI network layer 3, TCP to layers 4 and 5.

Contents

Figures

Tables

Introduction

The Network Executive Software NETwork EXecutive (NetEx®) family of software products is used with network hardware to enable two or more application programs (which may be on different host computers) to communicate with each other at multi-megabit speeds. The NetEx family of software consists of different versions of NetEx for use with different operating systems, such as this version for use with the Unisys 2200 Series. Network Executive Software also has utility programs available for use with NetEx, such as the Bulk File Transfer (BFX) utility, to simplify user programming to an even greater degree.

The NetEx software resides as a stand-alone real-time program within each host involved in the communication. As an independent program, NetEx allows communications to take place at any time during host operations, independently of other functions in the system. The following subsections describe the characteristics of NetEx and how NetEx uses the International Standards Organization guidelines for open systems interconnection.

NetEx Characteristics

NetEx centralizes network considerations for IP networks into a single piece of software. The following sections describe the characteristics of the NetEx software.

- External interface
- Internal interaction
- NetEx connections
- Design flow efficiency and flexibility
- Block segmenting
- Alternate Path Retry
- Basic I/O flow
- Remote operator interface

External Interface

The NetEx external interface for the application programmer is common for all versions of NetEx. NetEx provides requests for use in the programs that call NetEx. These calling programs may be written in C or other high-level languages. NetEx programs written in high-level languages may be transported from one host to another, with some changes to account for different word sizes and other machine architecture variations.

NetEx also provides an operator interface that monitors and controls certain NetEx functions.

Internal Interaction

The internal operation of all supported versions of NetEx is consistent and allows the different versions to interact freely. Thus, any program using NetEx may communicate with any other program on the network that is also using NetEx.

NetEx Connections

To communicate using NetEx, two calling programs first form a connection using a handshake protocol. NetEx then allows this pair of programs to communicate.

NetEx can establish multiple connections at one time, and can allow one program to have multiple connections simultaneously.

NetEx also supports communications within a single host. A calling program may connect to another calling program in the same host and exchange information just as if network communications were taking place.

Design Efficiency and Flexibility

The NetEx design enables many applications on the same processor to share the use of the network facility. Programs calling NetEx can be written without regard to the other programs calling NetEx or other Network Executive Software device drivers.

Once NetEx accepts data from the caller, NetEx must deliver the data to its destination. The NetEx subsystem on each host handles flow control, error recovery. NetEx optimizes data transfer throughput using a high degree of parallelism. That is, under normal circumstances, simultaneous adapter I/O, NetEx buffer management, and user file I/O all take place concurrently. This means that the effective data transfer rate is as fast as possible (in the multi-megabit range).

Block Segmenting

NetEx products provide block segmenting at the transport layer. NetEx divides data into segments of a specified size for transmission across the network and reassembles the segments on the remote host before delivering the data to the session layer calling program on the remote NetEx. This segmenting is transparent to the session user but provides control of the transmitted block segment size.

Alternate Path Retry

Alternate Path Retry (APR) provides the capability for connections to automatically reroute on different network paths when a failure on a path is detected. This rerouting takes place with no loss of data. Alternate path retry is provided as part of the type 2 protocol supplied with current NetEx versions. For more information on APR, refer to the *"C" Configuration Manager and NetEx Alternate Path Retry (APR) User Guide*.

User Exits

NetEx provides user exits at well-defined points within the NetEx subsystem for security and accounting purposes. These exits are essentially "do-nothing" routines that may be replaced by user modules at installation time. User exits exist before each request starts processing and after processing is complete. It should be noted that, because of the wide variety of user needs, NetEx does not create generalized security and accounting information, but allows user exits to provide for these needs.

Remote Operator Interface

This version of NetEx provides a remote operator interface that allows users to issue NetEx operator commands to other defined NetEx hosts on the network. Other users may also be the remote operator for this NetEx. See "REMOTE Command" for more information. Security features are provided.

Basic I/O Flow

Figure 1 shows the basic I/O flow between two programs using host based NetEx. The calling program communicates with NetEx through the NetEx user interface. NetEx then uses the available network hardware to communicate with the calling program on the other processor.

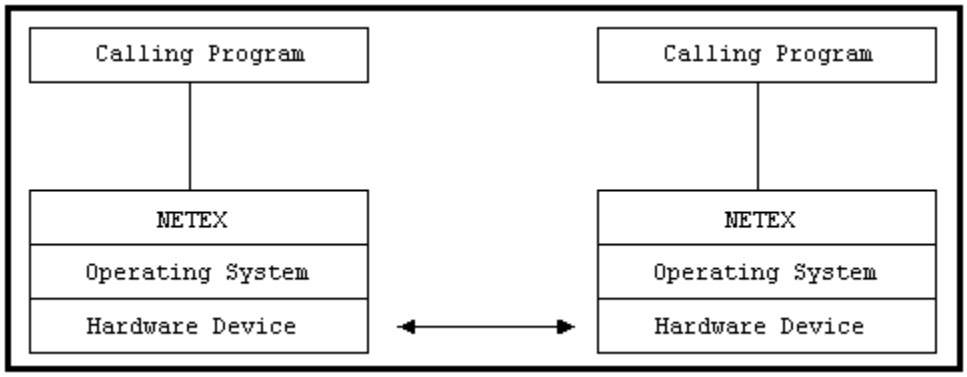


Figure 1. Basic I/O Flow

Host Based NetEx

Host based NetEx is an architecture that is designed for implementation on very large mainframe computers or on some smaller machines that cannot support the creation of a standard Network Executive Software driver product. Host based NetEx exists on the machine as a subsystem (a separate program residing in a machine that all other users in the machine can call on to perform services). User tasks produce a NetEx request that is delivered to the independent NetEx program using an inter-task communications facility provided by the host operating system. Data is moved so it is present in the NetEx program and the I/O is performed in the NetEx program.

Host based NetEx provides an administrative capability to the system programmers and system managers. Since all I/O is performed by the NetEx program, no data can be introduced on the network without first being checked by NetEx.

Host based NetEx products are implemented in 'C', Assembler, PASCAL, and other languages.

IP NetEx

NetEx/IP is a host based NetEx that supports IP networks. It interfaces directly with an IP NIC driver (Unix, IBM OS390 or Unisys OS2200 systems.

NetEx and the ISO Model

In creating NetEx, Network Executive Software followed the guidelines set by the International Standards Organization (ISO) for Open Systems Interconnection (OSI). Open Systems Interconnection refers to the exchange of information among terminal devices, computers, people, networks, etc., that are open to communication with one another.

The ISO model is composed of seven layers. Each layer interacts only with adjacent layers in the model (see Table 1). By using this modular structure, the internal function of each layer is self-contained and does not affect the functioning of other layers.

Table 1. ISO Model

Layer	Major Functions
Application	High level description of data to be transferred and the destination involved
Presentation	Select data formats and syntax
Session	Establish session connection, report exceptions, and select routing
Transport	Manage data transfer and provide NETEX-to-NETEX message delivery
Network	Point-to-point transfer, error detection, and error recovery
Data Link	Data link connection, error checking, and protocols
Physical	Mechanical and electrical protocols and interfaces

Although each layer physically interacts only with adjacent layers, each layer appears to communicate directly with the corresponding layer of the other model. Figure 2 illustrates this concept.

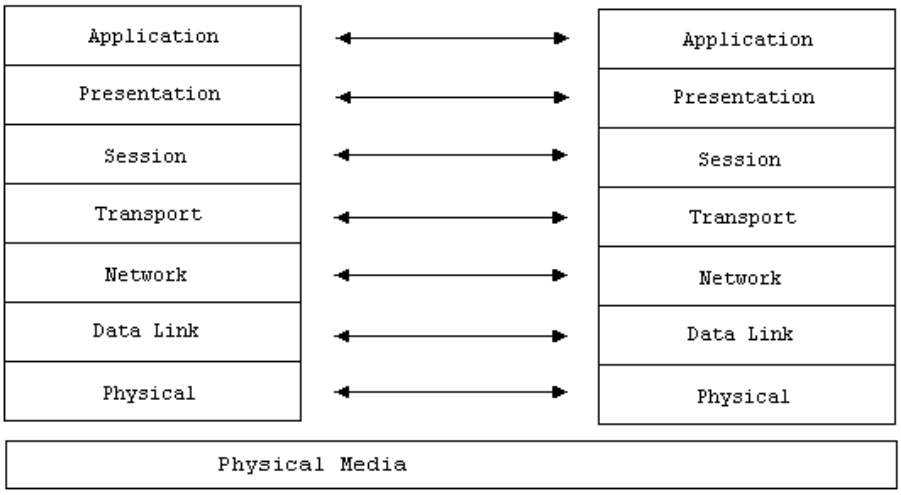


Figure 2. ISO Model Communication

Note: The corresponding layers appear to communicate directly as indicated by the lines with arrows, but actually they communicate only by progressing down through the layers of one model, through the physical media, and up through the layers of the other model.

Figure 3 shows that the hardware and firmware form the lower two layers. NetEx and the driver comprise the next three layers. The NetEx software provides complete session, transport, and network layer interfaces. This leaves the user free to write the application programs that use NetEx or to use Network Executive Software utilities.

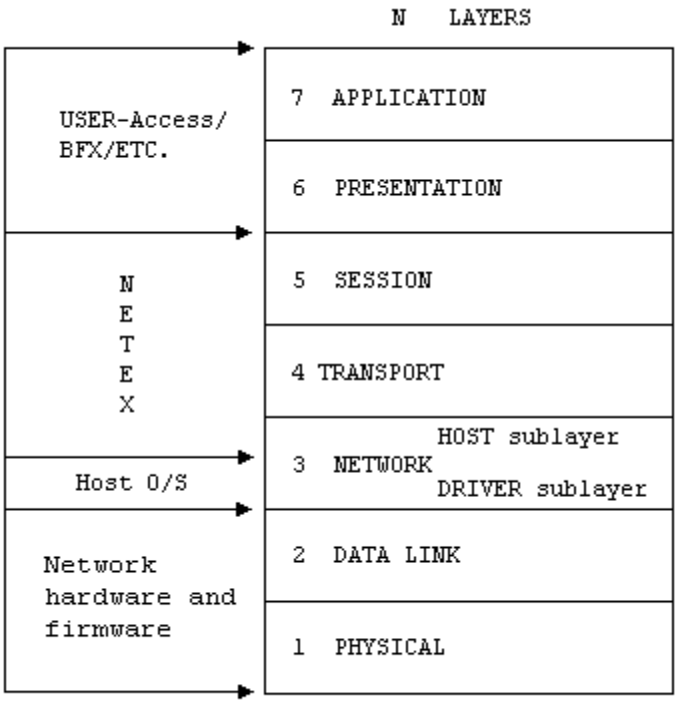


Figure 3. NETEX and the ISO Model

Session Layer Services

As the highest layer within NetEx (referring to the ISO model in Figure 3), the NetEx session layer software provides the general interface to the user's application/utility program. The NetEx session layer services include: program-to-program connection using the best available network path, reading data, writing data, disconnection, and statistics gathering. The user requests these services using a standard NetEx Request Block (NRB) (containing parameters), and the NetEx requests described in "NetEx Session Services". The session layer software implements user requests by requesting services from the underlying transport layer.

Transport Layer Services

The transport layer provides the actual data movement services for NetEx. This is an internal layer used only by the session service code, not the end user. It transmits and receives user data, along with internal protocol information, to provide fast, efficient communications over the network. The transport layer accomplishes its function by performing services for the session layer software above it and by requesting services of the network layer below it.

The transport software manages the network path chosen by the session software. The session user does not need to be concerned with the actual hardware and software used to transmit data, nor with NetEx-to-NetEx message delivery. The transport layer sets up hardware and software tables, provides buffering,

and establishes linkages to manage the flow of information. Also, the protocol used by the transport layer software provides true full-duplex communications between subsystems, permitting asynchronous reads and writes. Because the transport layer provides a full-duplex operation, data can flow continuously, as long as it is being supplied by the user. This keeps the communications link as busy as possible and assures timely arrival of data to the user.

Network Layer Services

The network layer software provides link independence for the higher layers of NetEx and assumes responsibility for keeping the network interfaces busy. This is an internal layer used only by the internal transport service, not the end user. The network layer formats the message proper to route the data through the network. If the protocol information overflows the HYPERchannel message proper, the network layer splits the data transmissions into two driver requests. The network layer also multiplexes network connections over common driver connections and manages those driver connections.

Driver Sublayer Services

The driver sublayer software is the interface between the network sublayer and the physical network device. The driver converts network sublayer I/O for a particular network path into a form which is understandable to the devices. The driver delivers and receives network messages and associated data to and from the network adapters. The driver also allows retry and error recovery for network adapters, supports assembly/disassembly, and code conversion options, if these are provided by the adapter type and requested by the user's data mode parameter.

Understanding Intertask Communication

General

The application programmer uses the NetEx subsystem for performing intertask communication. Like other subsystems, users can call upon the NetEx subsystem to perform services. NetEx must reside in the host of each task that is communicating.

To communicate using the session layer of NetEx, the calling programs must first establish a session connection. Once the session is established, data transfer may take place in a variety of ways, depending on the needs of the calling programs. NetEx uses internal error checking and error recovery procedures that are transparent to the user. Once NetEx accepts data, the user is assured of its delivery of the packets (except in the case of catastrophic failures - for example, a machine going down or a major problem with the communication line or the other program going away). Either party may terminate a session at any time, although this should be done by mutual agreement.

This section explains the concepts of intertask communication using NetEx. The section discusses the following topics:

- Session Layer Requests
- How a NetEx Session Works
- Network Communication
- NetEx Error Recovery Procedures
- Code Conversion

Session Layer Requests

There are eight active requests used by calling programs to call NetEx at the session level. These requests must be issued in a logical order (according to rules described in the following paragraphs). These eight requests and a table called the NetEx Request Block (NRB) are the calling program's interface to NetEx. The calling program updates the NRB when the program issues requests, and NetEx updates the NRB when requests are completed. The NRB is completely described in "Understanding the NetEx Request Block" on page 27.

Session layer requests may be issued in Assembler or FORTRAN. The following sections provide examples of each language's request format:

- "Assembler Interface" on page 41
- "FORTRAN Interface" on page 53

The functions of each of these language requests are the same and are discussed in general terms in this section. Rules for using the requests are also provided in this section.

The active NetEx requests (listed in the approximate order in which they are issued in the calling program) are described below.

OFFER

This request makes a program calling NetEx available to another program on either a remote or local host.

CONNECT

Requests a session with a calling program that previously issued an OFFER. The program may insert values defining characteristics of the upcoming session in the NRB when this request is issued. This request may also write data to the OFFERing program.

CONFIRM

This request is the last step to establish the session. The host that initially issued the OFFER replies to a CONNECT with the CONFIRM request if the characteristics of the session (for example, data block size) specified in the CONNECT are accepted.

READ

This request receives data that has been written by another calling program. The READ request also accepts indicators such as CONFIRM, and DISCONNECT. The READ request is an asynchronous service, so the user may continue when NetEx accepts the READ request.

WRITE

This request sends data to the other program. The WRITE request may only be used after a session has been properly established. The WRITE request is an asynchronous service. The user is free to continue when NetEx accepts the WRITE. A datamode may be specified to invoke code conversion and data assembly or disassembly.

WAIT

This command is specified as part of a request or as a separate request. WAIT suspends processing on the issuing program until NetEx completes a specific request or one of a list of requests. In the case of WRITE, CONNECT, CONFIRM, CLOSE, or DISCONNECT requests, NetEx accepts the request quickly and the issuing program can consider the request complete. In the case of READ and OFFER requests, the request does not complete until the other program issues a WRITE, CONNECT, CONFIRM, CLOSE, or DISCONNECT request, or the read-timeout value is reached.

CLOSE

This request is usually the last write operation for a connection. The CLOSE request gracefully terminates a connection. It may contain data just like a WRITE request, but it also indicates that the sender is ready to terminate the connection. After the CLOSE is issued, incoming data may continue to be read from the session the CLOSE was issued to, but no other messages may be written. When the other party in the connection issues a CLOSE, the session is gracefully terminated.

DISCONNECT

This request immediately terminates a connected session. The DISCONNECT request may be issued by either program at any time. The DISCONNECT may also withdraw OFFERs.

These requests are used during the session as described in the following sections.

How a NetEx Session Works

This section explains how a simple NetEx session works.

Figure 4 shows a simple example of a session. The program calling NetEx, known as a calling program, reads data from another program.

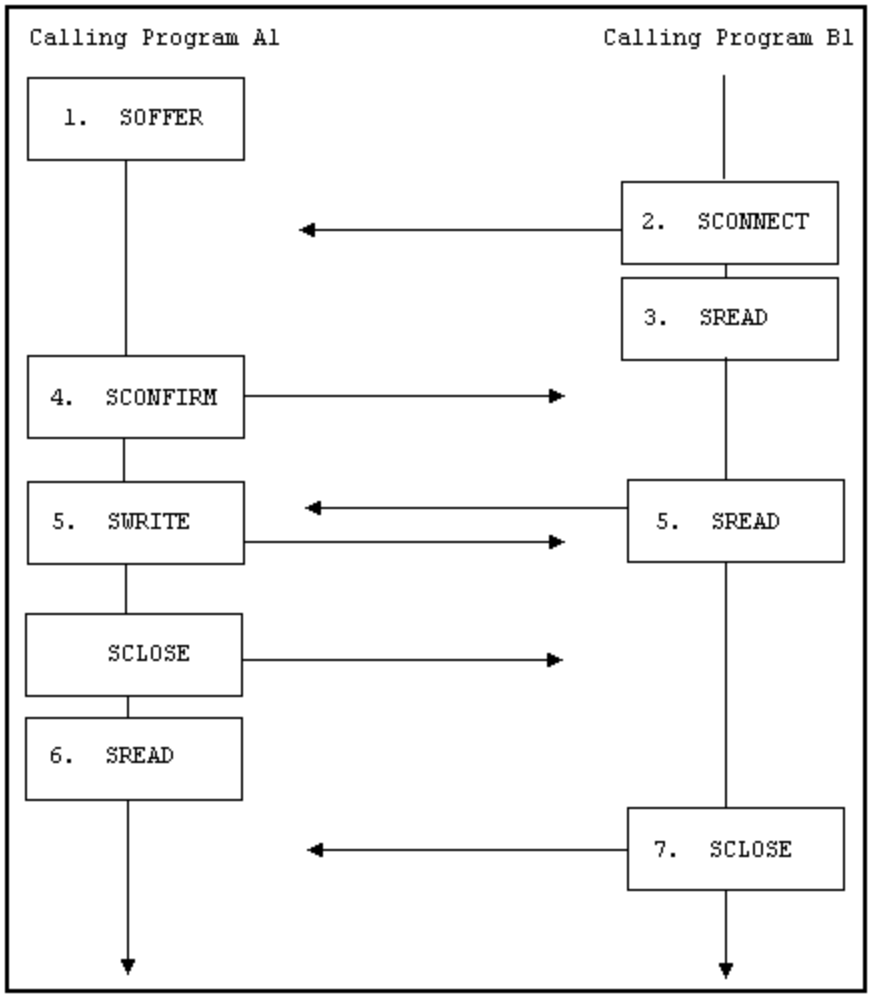


Figure 4. Two Programs Conducting a Session Using NetEx

The following text describes the session flow shown in Figure 4:

- Calling program A1 in host A issues an OFFER indicating that it is available to service other calling programs.
- Calling program B1 requires a file controlled by program A1. To initiate a data transfer session, program B1 issues a CONNECT request. This request may contain data to be delivered to the OFFERing program.
- When the CONNECT completes (when it is accepted by NetEx), program B1 issues a READ and prepares to receive program A's response to the CONNECT.

- When the NetEx in calling program A1's host receives the CONNECT, the OFFER completes with a connect indication and with B1's CONNECT data in the buffer associated with A1's OFFER. If program A1 finds the conditions associated with the CONNECT acceptable, it responds by returning a CONFIRM request.
- The programs can begin transferring data. Program B1 issues a READ to prepare to receive data from program A1. Program A1 writes data to program B1. Program A1 uses WRITE command until the last data is written. A CLOSE writes the last data. Since we are only issuing one write request in this example, the CLOSE is used.
- After the last data transfer completes, program A1 issues a READ to detect program B1's next request.
- Program B1 issues a CLOSE and the session is terminated. Both programs can perform disconnect functions (for example, closing files). Program A1 can now issue an OFFER to declare itself ready for another session.

This described a simplified example of a session. The following sections describe how to program NetEx by describing the following topics in detail:

- Establishing a Session
- Data Transfer
- Terminating a Session
- Handling Multiple Connections
- Withdrawing OFFERs

Establishing a Session

The flow chart in Figure 5 shows how to establish a connection using the session layer interface. Only steps that may occur in a normal process are shown Figure 5. The following text discusses other possibilities that are less likely to occur.

Figure 5 refers to the NRB. The NRB (discussed in "Understanding the NetEx Request Block" on page 27) is a block of parameters used to signal requests to NetEx and to return status to programs.

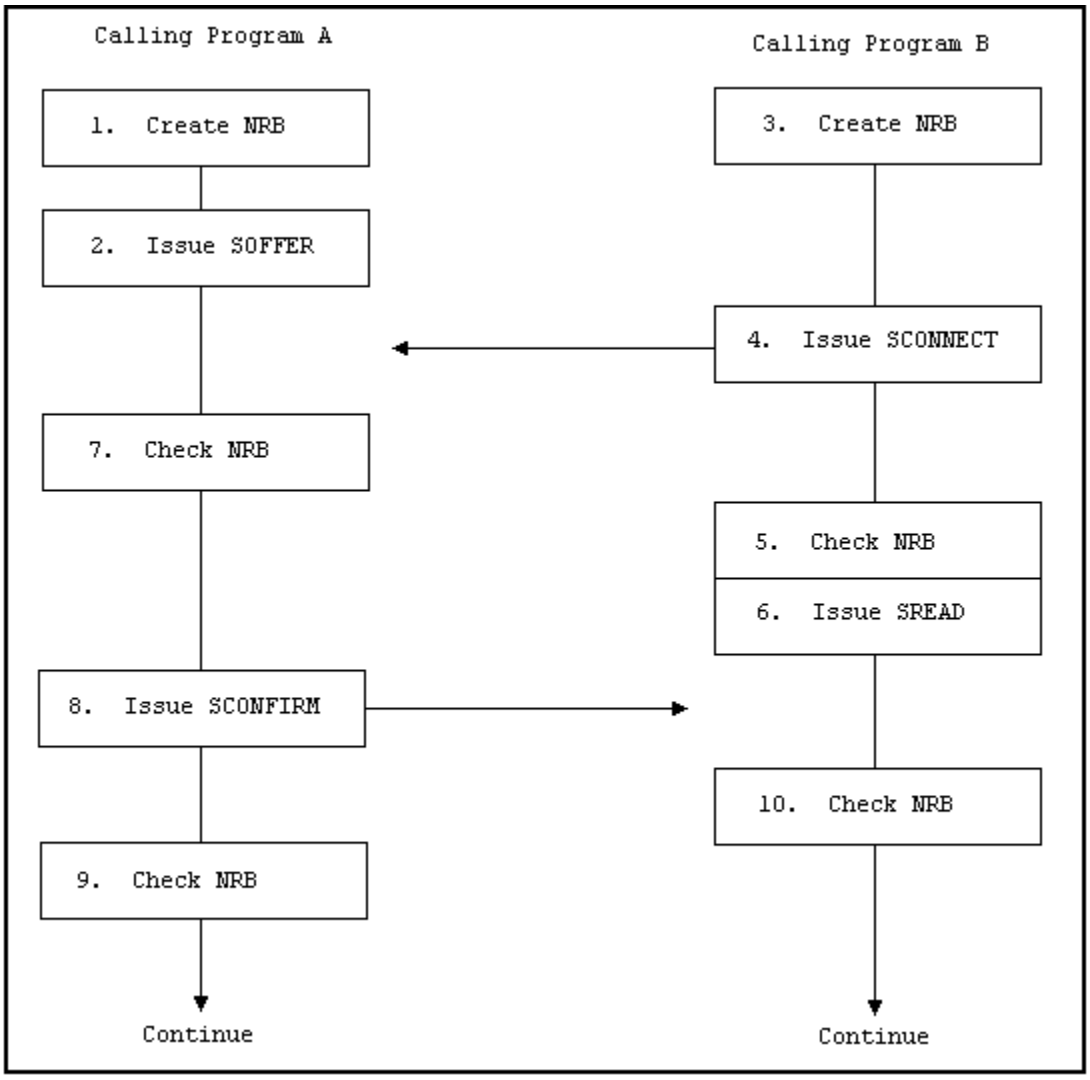


Figure 5. Establishing a Connection

The following numbered items refer to the steps in Figure 5. The steps are numbered to simplify the discussion and do not necessarily represent the exact order in which the events occur.

1. Program A prepares for the session by opening files and creating an NRB.
2. Program A issues an OFFER to make it available to other NetEx programs. The OFFER may specify a data area for data associated with an upcoming CONNECT.
3. Program B needs to establish a session with program A. Program B must first open files and create its own NRB. If program B had previously issued an OFFER, it can still issue a CONNECT provided it uses a different NRB. However, program B must have some provision for responding to (or ignoring) CONNECTs that are issued against the outstanding OFFER.
4. Program B issues a CONNECT. The CONNECT may contain data such as a password.
5. Program B checks the NRB to determine the status of the request. The NRB indicates if a request is in progress, if a request has completed successfully, or if the request has generated an error.

   Figure 5 continues assuming the NRB indicated normal completion. If the NRB indicates that a request is in progress, it would have to be rechecked after a short delay. If the NRB indicates an

error, the error code would be logged and the session would not be established. The calling program should try again to establish a session or take appropriate action. For example, closing files that were opened before the session was attempted.

6. Program B expects program A to respond to the CONNECT with a CONFIRM or a DISCONNECT. Program B issues a READ that detects program A's response.
7. Program A checks its NRB to see whether the OFFER completes. The NRB indicates that program B has issued a CONNECT.

   If the NRB indicates that an error occurred, program A takes appropriate action, such as disconnecting from this session and reissuing the OFFER.

   A password may be required at this time. The password could be sent as data associated with the CONNECT. After the CONNECT is received, the password is examined. The password can restrict access to certain files, restrict access by certain programs, or have other customized uses. If a program attempts to access restricted files or has an incorrect password, program A may issue a DISCONNECT and terminate the session.
8. If all conditions associated with the CONNECT are acceptable, program A issues a CONFIRM to establish the session. The CONFIRM may contain data.
9. Program A checks the NRB to make sure that the CONFIRM was accepted by NetEx. If it was, program A would continue with the session. If NetEx did not accept the CONFIRM, program A would either retry issuing the CONFIRM or take other appropriate action.
10. Program B checks the NRB to determine if the READ has successfully completed and to see what call program A issued. If program A had responded with a CONFIRM, program B would continue with the session. If program A had responded with a DISCONNECT, program B would have to examine the reason code associated with the DISCONNECT and take the appropriate action.

The previous discussion outlines the rules to be followed when establishing a NetEx session. It also introduces the concept of using the NRB for communication with NetEx. After requests are issued, the NRB is examined to see when NetEx completes the request. This may be done using the WAIT request, or by periodically checking the NRB fields. The NRB fields are discussed in "Understanding the NetEx Request Block" on page 27.

Data Transfer

NetEx provides a great deal of flexibility in how session requests are used for the data transfer process. The following sections describe three methods for programming data transfer:

- Write/Read Data Transfer
- Concurrent Write and Read Data Transfer
- One-Way Data Transfer

Write/Read Data Transfer

The following paragraphs describe the session requests used to transfer data in a straight-forward way. In general, calling programs use the READ and WRITE requests to perform the data transfer. Figure 6 shows how the calling programs perform the data transfer. WRITE requests are issued by one program that must be received by a READ issued by the other program.

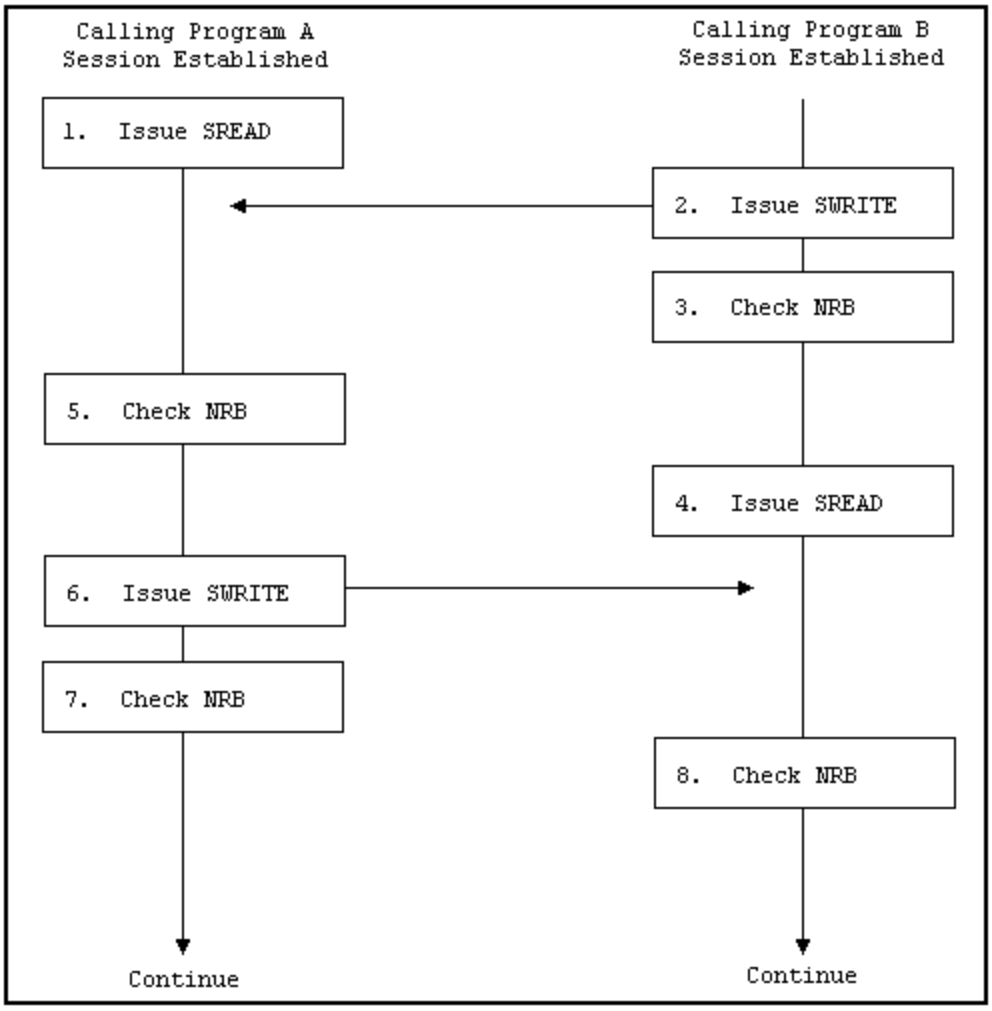


Figure 6. Write/Read Data Transfer

The following list describes the steps in Figure 6. The steps are numbered to simplify the discussion and do not necessarily represent the exact order in which the events occur.

1. After a session has been established, program A issues a READ. The READ specifies the buffer for receiving data.
2. Program B issues a WRITE. The WRITE specifies where the data to be written is located and data length.
3. Program B checks the NRB to see if NetEx accepted the WRITE or indicated an error.

   Once NetEx has accepted the data, NetEx delivers the data unless a catastrophic loss of the connection occurs.
4. Program B issues a READ to detect program A's next request.
5. Program A received an updated NRB that indicates what program B has issued. In this case, program B has written data as program A expected. If the NRB word indicated an error, program A takes appropriate action.

   If program B issued a DISCONNECT, program A would check the reason for the DISCONNECT and take appropriate action.

6. Program A is programmed to WRITE some data back to program B. Program A issues a WRITE specifying the location and length of the data to be written.
7. Program A verifies that NetEx accepted the WRITE by checking the NRB. If the NRB indicates the WRITE was not accepted, program A would take appropriate action.
8. Program B checks the NRB and determines what program A issued.

   Both programs continue with the data transfer until they have completed their functions.

As when establishing a session, the WAIT request may be used with other requests. Abnormal terminations are discussed in "Abnormal Session Termination" on page 22.

The following sections contain examples of calling programs:

- "Assembler Interface" on page 41
- "FORTRAN Interface" on page 53

Concurrent Write and Read Data Transfer

Issuing READ and WRITE requests without expecting the other program to respond immediately is an advanced method of data transfer. Use this technique for network communication (see "Network Communication" on page 24 for more information). It is also well-suited for local data transfer.

Because NetEx only accepts one request using a specific NetEx Request Block (NRB), each calling program must create two NRBs to perform concurrent READ and WRITEs. One NRB establishes the session (as described in "Establishing a Session" on page 13) and a second is created before data transfer begins. The second NRB must be created as a copy of the first to ensure that NetEx internal words are preserved.

Figure 7 shows how the calling programs perform the data transfer. Program A first requests data from program B. Program B then WRITEs data until program A WRITEs an acknowledgment or another message. Notice that program A does not respond to every WRITE issued by program B. When program A has received what it needs, it terminates the session using the termination procedure discussed in "Terminating a Session" on page 21.

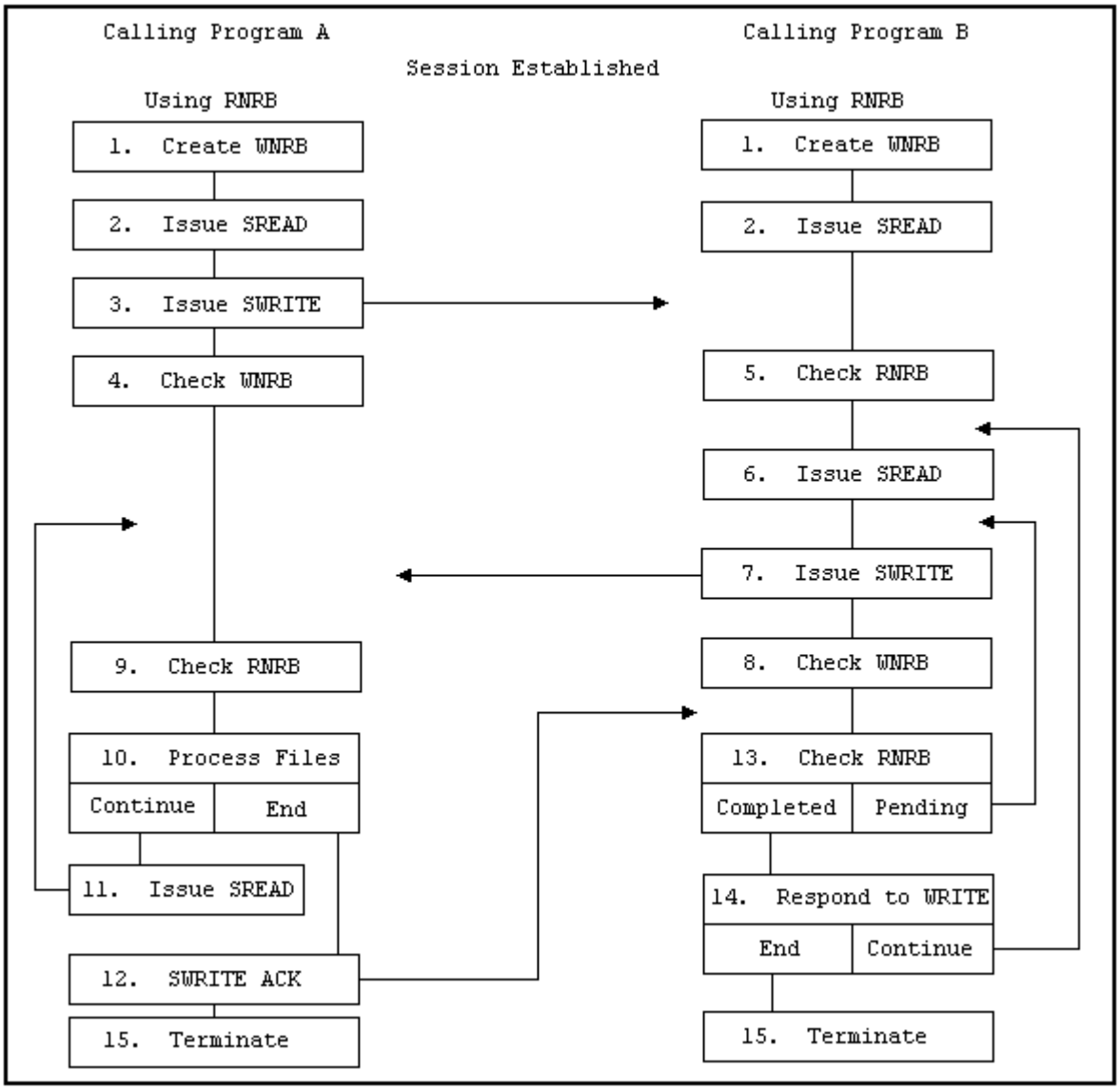


Figure 7. Concurrent SREAD and SWRITE Requests

The following list describes the steps in Figure 7. The steps are numbered to simplify the discussion and do not necessarily represent the exact order in which the events occur.

1. After a session has been established, both programs create duplicate NRBs for the WRITE requests. The NRBs created before the sessions were established are assumed to have been called RNRB, and are used with READ requests. New NRBs called WNRB are duplicates of the RNRB used with the WRITE requests
2. Both programs issue a READ request to prepare to receive requests from the other program. The RNRB is specified in the READs as the place for NetEx to respond to that request.
3. Program A issues a WRITE to program B. The WRITE contains a request for specific data from program B. The WNRB is specified in the WRITE as the place for NetEx to respond to that request.
4. Program A checks the WNRB to see if NetEx accepted the WRITE or indicated an error, and acts accordingly.

5. Program B, which has been checking the RNRB or WAITing for it to complete, receives the WRITE from program A. This WRITE contains parameters that program B uses to determine what data to send to program A.
6. Program B issues another READ that floats while processing continues.
7. Program B begins to WRITE the data requested by program A. The WNRB is specified to monitor the WRITE requests.
8. Program B checks the WNRB to make sure NetEx accepted the WRITE.
9. Program A checks its RNRB and discovers the WRITE issued by program B.
10. Program A processes the files received. If program A has not yet received all the data it asked for in step 3 and wishes to continue READing, it jumps to step 11. If program A wishes to respond to program B (to stop the transfer or to request other data), it jumps to step 12 and WRITEs an appropriate message.
11. Program A issues a READ to continue receiving information from program B.
12. Program A WRITEs an acknowledgment or a message to program B. Since program B has a READ floating, it receives this WRITE.
13. Program B checks the RNRB. If the READ has completed (meaning program A has written something), program B continues with step 14. If the READ is still pending (or floating), Program B continues WRITING data to program A by jumping back to step 7.
14. Program B responds to program A's WRITE. This response could include starting to transmit other data, receiving an acknowledgment, recording a message, and terminating the session.
15. The session is terminated normally when program A has received all the data it wanted, or by special request of one of the programs. Session termination is described in "Terminating a Session" on page 21.

The previous example demonstrates the technique of using READ and WRITE requests concurrently. WAITs should only be used after WRITE requests when using this technique. Abnormal terminations are discussed in "Abnormal Session Termination" on page 22.

One-Way Data Transfer

A typical use of NetEx is a one-way data transfer. Figure 8 shows how a one-way data transfer could take place. Calling programs A and B establish a session (as described in "Establishing a Session" on page 13). Program A, that receives data, creates a single NRB. Program B, that sends data, creates an RNRB for monitoring READ requests and a duplicate WNRB for monitoring WRITE requests.

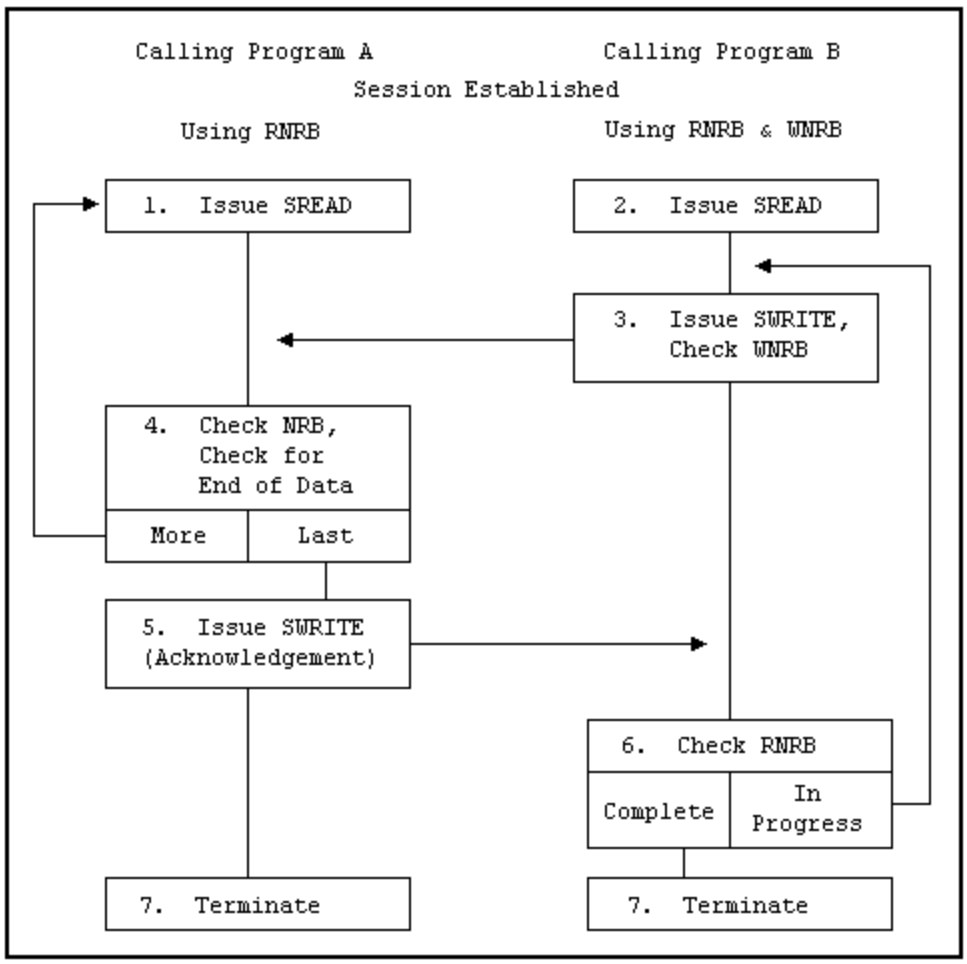


Figure 8. One-Way Data Transfer

The following list describes the steps in Figure 8. The steps are numbered to simplify the discussion and do not necessarily represent the exact order in which the events occur.

1. Program A issues a READ request to prepare to receive data.
2. Program B issues a READ request to receive unexpected WRITEs from program A. For example, if program A were unable to receive more data, it could notify program B using a previously determined set of indicators. Normally, this READ would not complete until all the information has been transferred.
3. Program B issues a WRITE to transfer data to program A. An end of data indicator is placed in the data field with the last piece of information. Program B checks the WNRB to see if NetEx accepted the WRITE or indicated an error, and acts accordingly.
4. Program A checks the NRB to see if the READ completed. If it did not complete, program A continues to check it. When the READ completes, program A processes the incoming information, and checks for the end of data indicator. If there is more data coming (indicator not set), program A issues another READ (step 1). If this is the last piece of information, program A continues with step 5.
5. Program A issues a WRITE acknowledging that all of the information has been received. (NetEx has verified the integrity of the data.)

6. Program B checks the RNRB. If it is still in progress (because program A has not written anything), program B jumps back to step 3 and WRITEs more data. If all data has been written, program B issues a WAIT to suspend execution until program A returns a response. When the RNRB completes, program B checks the data written by program A. Normally, this would be an acknowledgment of the last piece of data. In that case, program B would continue with step 7. If a problem is encountered, program B acts accordingly.
7. Program B terminates the session. Terminating a session is described in "Terminating a Session" on page 21.

In the previous example, WAIT requests may be used freely by program A, but should generally be used only after WRITE requests by program B. A WAIT could be issued by program B to wait on the RNRB after all data has been written.

Terminating a Session

NetEx sessions can terminate either normally or abnormally. A normal termination is planned by the programs involved. An abnormal termination is any unplanned termination of the session. See the following sections for more information:

- Normal Termination
- Abnormal Session Termination

Normal Termination

Figure 9 shows the exchange of session calls associated with normal (planned) termination. The steps are numbered to simplify the discussion and do not necessarily represent the exact order in which the events occur.

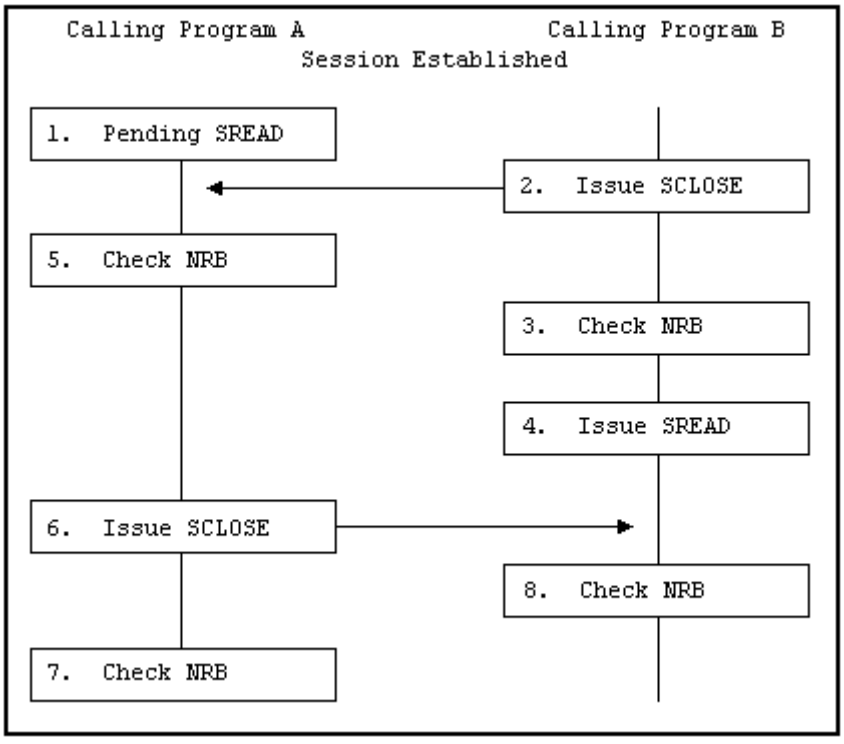


Figure 9. Normal Session Termination

The following list refers to the steps in Figure 9.

1. Program A has a previously issued READ pending.
2. Program B issues a CLOSE. The CLOSE takes the same form as a WRITE request.
3. Program B checks the NRB to verify that the CLOSE was accepted by NetEx. If it was accepted, program B may close output files or perform other termination processing. No other NetEx write-type requests (except DISCONNECT) may be issued by program B during this session. If the CLOSE is not accepted, Program B must take appropriate action (check if NetEx is down or if the other application is not there).
4. Program B issues a READ. Program A may still write data to program B, or program A may issue a CLOSE or a DISCONNECT to terminate the session.
5. Program A detects the CLOSE.
6. Program A issues a CLOSE to terminate the session. Data may be associated with this request. Program A may issue any number of WRITE requests before the CLOSE.
7. Program A checks the NRB to make sure that the CLOSE completed successfully. Program A closes files and performs other termination processing.
8. The READ issued by program B completes and the NRB indicates that the session has been terminated. Program B closes files and performs other termination processing.

Abnormal Session Termination

The calling program must react to abnormal terminations. Sessions may be abnormally terminated by the other program or by NetEx.

Sessions may be unexpectedly terminated by the other program for various reasons (depending on how the program is written). Some typical reasons for immediate termination are listed below:

- A calling program fails to provide the proper password or authorization for a session.
- A calling program attempts to access data that it was not authorized to access.
- The calling program detected an internal failure such as a program check.
- A time limit was reached.
- A calling program encountered problems issuing a request to NetEx.

Some of these problems may be overcome by reconnecting with the calling program.

NetEx may terminate a session unexpectedly because of problems with the physical network or with a host computer. This type of error may not necessarily be solved by simply reconnecting with this host. Alternatives should be provided in the calling program.

Handling Multiple Connections

NetEx provides the capability for a calling program to be simultaneously connected to more than one other calling program. Requests coming from different connections are identified using the NRBNREF word of the NRB. NRBNREF contains a unique number assigned to a session connection when it is established. The programmer can use NetEx's ability to handle multiple connections to establish database server and requestor applications.

Database server applications allow a network of hosts to use each other's databases. A database server application simply OFFERs itself to other hosts. When another host (a requestor) establishes a connection, the server READs or WRITEs files to or from its database as specified by the requestor.

Database server applications may issue multiple OFFERs by specifying a different NRB with each OFFER. The OFFERs are completed in the order that they are issued.

Program B in the concurrent WRITE/READ example (Figure 7) is an example of a simple database server. Program B WRITEs the files program A requests, and then waits for more instructions. More sophisticated database servers could allow themselves to connect to several requestors at one time.

Program A in Figure 7 is a simple example of a database requestor. File 7 of the NetEx install tape also provides sample programs.

Withdrawing OFFERs

A user may need to selectively withdraw an OFFER that has been presented to NetEx (especially when using multiple connections). The DISCONNECT command may be used to perform this withdrawal, but there are two different ways to use the DISCONNECT. One method withdraws all OFFERs outstanding; the other method withdraws only a specified OFFER.

To withdraw all OFFERs, copy an NRB used with one of the OFFERs and zero-fill the NRBSTAT and NRBNREF fields. This terminates all OFFERs.

To withdraw a specific OFFER, copy the NRB used with the OFFER and zero-fill the NRBSTAT field. This terminates only the specified OFFER.

Network Communication

Network communications facilities can use standard NetEx requests when they are a part of the network. NetEx uses protocols that are transparent to the user to recover from errors and lost messages.

Communications Channel

Because of the relatively long propagation delay inherent in long distance networks, you must keep the communications channel full of data. Use concurrent multiple READs and multiple WRITEs that transmit data in large amounts before waiting for a response from the other calling program. In this way, data is transmitted rapidly, and the propagation delay has less effect on performance.

Example

If the calling programs acknowledge every block written in one direction with a corresponding acknowledgment written in the other direction, the propagation delay has a major impact. If an entire file is transmitted before an acknowledgment is returned, the effect of the propagation delay is minimized.

Minimizing the effect of the delay in this manner must be balanced with the following consideration: if there is a catastrophic failure of the link, NetEx, or the other host, determining how much unacknowledged data was successfully received is difficult.

Checkpoint Acknowledgments

Determine the frequency of the checkpoint acknowledgments. Consider the needs of individual implementations before making this decision. The RATE and DELAY parameters on the Configuration Manager PORT statement, or the corresponding ROOT macro, determine the most efficient transmit scheme.

NetEx Error Recovery Procedures

Calling programs must be able to recover from errors identified by NetEx. These errors are returned when a NetEx operation does not complete successfully. The following sections describe the NetEx error codes and some common error recovery procedures:

- Error Codes
- Common Error Recovery Procedures

Error Codes

Whenever a NetEx request is issued, the results of the request are returned in one or both of two NRB fields, NRBSTAT and NRBIND. These are located at the beginning of the NRB to make their subsequent examination by high level language programs a simpler matter.

NRBSTAT indicates if an operation is in progress and whether it completed successfully or not. NRBIND indicates the type of information that arrived as the result of a read-type command (READ or OFFER).

When an operation is accepted by the NetEx user interface, the value of NRBSTAT is set to the local value of -1. The sign of this word functions as an operation in progress flag for all implementations.

If an operation completes successfully, NRBSTAT is returned as all zeroes. If a read-type command was issued, then an indication is set in NRBIND when the READ completes.

If the operation did not complete successfully, then NRBSTAT contains a standard error code. The error codes are described in the section “Appendix A. NRBSTAT Error Codes”.

Common Error Recovery Procedures

The following list describes commonly encountered errors and explains how to recover from these errors.

- Other application is not there. The running of the two NetEx programs must be coordinated so that one has not timed-out before the other NetEx program establishes a session.
- Other application is busy. Retry NetEx after a suitable delay.
- NetEx requests are out of sequence. Sessions must be completely established before WRITE or READ requests can be issued. Sessions are established using the OFFER, CONNECT, and CONFIRM requests (in that order).

Code Conversion

NetEx provides for common types of code conversion by using NetEx software facilities. The calling program uses the datamode (NRBDMODE) word of the NRB to specify either manual or automatic code conversion.

Manual Code Conversion

The caller specifies the assembly/disassembly and code conversion functions for both output and input. The caller must determine which assembly/disassembly modes and code conversion tables are meaningful.

Automatic Code Conversion

The caller specifies only the source character set and the destination character set. NetEx uses software code conversion when necessary.

Understanding the NetEx Request Block

General

The NetEx Request Block (NRB) is a block of parameters used to pass information between calling programs and NetEx. The NRB is the means by which programs and NetEx communicate with one another. The NRB is created by a calling program and may be updated by the program to pass information to NetEx, or NetEx may update the NRB to pass information to a program.

Each time a program makes a request to NetEx, the program specifies an NRB to be associated with the request. NetEx passes status information about that request back to the program by way of the NRB. Therefore, only one NetEx request may use an NRB at one time. If concurrent read and write requests are used, or if a server program will be connected to more than one program at a time, several NRBs must be used.

The use of the NRB fields varies slightly between the different levels of programmer interface. Specific differences are described in the individual field descriptions that follow.

This section discusses the following topics:

- Understanding NRB Words
- Creating an NRB
- Duplicating an NRB

Rules for NRB Use

The following principles are designed to make high level language use of NetEx somewhat transportable between machines.

- Before initiating a connection, the user must clear the NRB, including the operating system dependent portion, to zeroes. When the connection is initiated, the user places whichever non-default values are needed in the user part of the NRB, and invokes NetEx service. Once the connection is initiated, the user must not change the OS dependent part of the NRB between calls to NetEx.
- If the calling program is using the connection in a full duplex manner, the user will need to make a copy of the NRB to produce a “read NRB” and a “write NRB.” This copying operation is the copying of all 40 fields of the NRB to another area, at a time when the NRB being copied is not active. If a second request for the same connection is issued from the copied NRB, the user interface must detect the condition and handle that new part of the connection accordingly.
- Many NRB values are specified in addressable units. An addressable unit is the amount of information contained in one memory location for that machine. For example, an addressable unit on Unisys OS2200 is 36 bits, IBM is 8 bits, Tandem is 8 bits, and so on.

Understanding NRB Words

Figure 10 shows the words in the NRB. Most NRB fields are one word long. The NRBPNAME and NRBHNAME fields are two fullwords long. The NRB contains forty 32-bit words (160 bytes) and must be aligned on a fullword boundary.

The calling program or NetEx can update many of the NRB words with every request. However, the words NRBCLASS, NRBMAXRT, NRBBLKI, NRBBLKO, NRBRESV, NRBPNAME, and NRBHNAME are associated with the session negotiation process. NetEx updates information in these fields as their values change. These fields are initially specified during the OFFER and CONNECT requests. When the Offering task receives the connect, the negotiated values are set in the OFFERed NRB. When the CONFIRM is sent, the negotiated values are set in the NRB associated with the READ of the CONFIRM information. Subsequent attempts to change these fields have no effect.

NRB fields may be referenced by MASM programs using the names shown in Figure 10 if the NXDEFS omnibus element is $INCLUDE'd. Otherwise, the NRB words must be referenced using the index numbers shown on the left side of Figure 10.

1	NRBSTAT	-	Status and error code returned to user
2	NRBIND	-	Data type indication from OFFER/READ/STATUS
3	NRBLEN	-	Length of data
4	NRBUBIT	-	Unused bit count
5	NRBREQF	-	Code for user's NetEx request
6	NRBNREF	-	Identifier of connection process
7	NRBBUFA	-	Starting address of user's buffer
8	NRBBUFL	-	Length of user's buffer
9	NRBDMODE	-	Datamode for WRITE request
10	NRBTIME	-	Request timeout in seconds
11	NRBCLASS	-	Class of service
12	NRBMAXRT	-	Maximum data rate permitted
13	NRBBLKI	-	Maximum buffer size for input requests
14	NRBBLKO	-	Maximum buffer size for output requests
15	NRBPROTA	-	User ODATA buffer address
16	NRBPROTL	-	User ODATA buffer length
17	NRBRESV1	-	Reserved
18	NRBRESV2	-	Reserved
19-20	NRBPNAME	-	Name of process to OFFER/CONNECT (double-word)
21-22	NRBHNAME	-	Name of host to connect to (double-word)
23	NRBRESV3	-	Reserved
24	NRBRESV4	-	Reserved
25-40	NRBOSDEP	-	Reserved

Figure 10. NetEx Request Block (NRB) Words

Describing NRB words

The following sections describe the NRB words shown in Figure 10.

- NRBSTAT
- NRBIND
- NRBLEN and NRBUBIT
- NRBREQF, NRBOPSRV, NRBREQ
- NRBNREF
- NRBBUFA
- NRBBUFL

- NRBDMODE
- NRBTIME
- NRBCLASS
- NRBMAXRT
- NRBBLKI and NRBBLKO
- NRBPROTA
- NRBPROTL
- NRBRESV1 and NRBRESV2
- NRBPNAME
- NRBHNAME
- NRBRESV3 and NRBRESV4
- NRBSSNM
- NRBOSDEP

NRBSTAT

NRBSTAT contains a summary of the request issued by the user. If the request is currently in progress, the entire field contains a -1 (all ones). If the request completed successfully, then NRBSTAT is 0. If the request was unsuccessful (NetEx or the service routine detected an error), NRBSTAT contains a binary representation of a decimal error code. See the "Appendix A. NRBSTAT Error Codes" section for explanations of these error codes.

The implementation user interface must be constructed so that a program polling NRBSTAT (to determine if the request was successful) immediately proceeds to examine the error code if a positive value is found in NRBSTAT.

A request is marked complete only after one of the following conditions is met:

- A WAIT option was integrated into the service call.
- An SWAIT request has been issued where one of the NRBs on the SWAIT list is the NRB specified.
- Any NetEx service call is issued, and NetEx service finds that the request has completed recently.

NRBIND

NRBIND indicates the type of data received in response to a read, offer, or status request. If any of those read-type requests are issued, NRBIND will always receive a non-zero value.

The values returned in NRBIND are:

(1) Connect Indication
(2) Confirm Indication
(3) Normal data Indication
(4) Expedited Data Indication
(5) Close indication
(6) Disconnect indication
(7) Status indication

If a write-type request (write, connect, confirm, close, or disconnect) is issued, the returned value of NRBIND is usually zero. If an error is returned to the write type request that means the connection is broken or was never established, then a Disconnect Indication (6) is set in NRBSTAT.

If an operation did not complete successfully, then NRBSTAT will be set to a positive, non-zero value. If NRBSTAT is non-zero, then NRBIND will have one of the following values:

- If the error results in the loss of the connection or the connection not being established in the first place, then a Disconnect Indication (6) will be in NRBIND.
- If the error means that the request could not be processed but the connection remains in effect, then NRBIND will be set to zero.
- If the data is "damaged" in input (for example, user buffer too small) then NRBIND will reflect the type of data received.

NRBLEN and NRBUBIT

NRBLEN and NRBUBIT together define the amount of useful data for input and output. NRBLEN specifies the number of addressable units (bytes for IBM) needed to contain the data. NRBUBIT specifies the number of bits in the last byte that are not significant information. This allows sending information on the network in a logical bit basis without damaging the data.

For example, suppose a Unisys computer wants to send exactly 35 of its 36-bit words to an IBM processor and wants it returned at a later date. The Unisys user will specify NRBLEN=35 and NRBUBIT=0. Datamode will be bit stream. NetEx will record that 35*36=1260 bits of information was sent over the network. The IBM user will receive the information with NRBLEN=157 (bytes) (8*157=1256 bits) and NRBUBIT=4 (bits). The IBM user could later specify the same length parameters on output and return precisely 35 words back to the Unisys.

A second example involving character conversion: Suppose a AIX wants to send 151 ASCII characters (8*151=1208 bits) to a Unisys and have them converted to Field-data. The AIX user can specify NRBLEN=19 (64 bit words) (64*19=1216 bits) and NRBUBIT=8 (bits) since seven characters will be in the trailing AIX word. The AIX driver will send the ASCII over the network and record that 8*151=1208 bits of information were sent. The Unisys will select sixth-word A/D mode and code conversion and determine that (1208/8)*6=906 bits of information will result. It will report to the Unisys caller that NRBLEN=26 (36*26=936) and NRBUBIT=30 so a single character will be found in the last of the 26 Unisys words.

Note: Those programs that do not need the NRBUBIT can ignore its existence, knowing that handling the information specified by NRBLEN will ensure that all information sent by the other machine will be stored or processed.

Transmitting or receiving zero-length information is possible. Zero-length data is treated as a separate transmission and is received at the other end in chronological order (as is any other data). On both the transmit and receive sides, NRBLEN will be set to zero.

If NRBUBIT is non-zero, the unused bits are not set to zero or any other value by NetEx. The calling program must handle any "garbage" that may be placed in the last word of the transfer.

NRBREQ

NRBREQ is the request code that will be given to NetEx. This is a 16-bit binary value that contains the type of request (example: SREAD) that NetEx is to perform.

NRBREQ has the following format:

Option Flags (4 bits)

The option flags refer to optional processing that NetEx will perform on the request. These flags are bit significant. The bits are assigned (represented as hexadecimal numbers) as follows:

0xxx	Normal processing. NetEx will return control to the caller when NetEx has internally queued the request.
1xxx to 7xxx	Reserved.
8xxx	WAIT. NetEx or the NetEx user interface is not to return control to the user program until the request is complete.
9xxx to Fxxx	Reserved.

Service Level (4 bits)

The service level indicates whether the request is a SESSION, TRANSPORT, NETWORK, or DRIVER type of request. The values are assigned (in hexadecimal) as follows:

x0xx	Session request
x1xx	Transport request
x2xx	Network request
x3xx	Driver request
x4xx to xExx	Reserved
xFxx	Reserved (effects Function values)

Function (8 bits)

Function indicates the specific type of request to be issued. The values are assigned (in hexadecimal) as follows:

xx01	Connect
xx02	Confirm
xx03	Write
xx04	Reserved
xx05	Close
xx06	Disconnect
xx07 to xx80	Reserved
xx81	Offer
xx82	Read
xx83	Status
xx84 to xxFF	Reserved

The total request code is produced by combining the Option, Function, and Service Level. For example, consider an SREAD with wait processing. Wait processing is 8xxx, SREAD is a x0xx Service Level plus a xx82 Function. This totals a 8082 (hexadecimal) request code.

NRBNREF

NRBNREF is the NetEx reference number for the connection. The value is assigned by NetEx when a connection is established. NRBNREF should be set to 0 on all Connect and Offer calls.

NRBBUFA

NRBBUFA contains the start of the data buffer used for either input or output requests.

NRBBUFL

On input, NRBBUFL specifies the maximum size of the information that NetEx can store in the user-specified read buffer. The user specifies this buffer size in the length parameter of the READ/OFFER command.

This field is ignored on output. NRBLEN and NRBUBIT determine the actual length of output data. This usage difference allows a NetEx user to associate an NRB with a single buffer and never change this field, even if many READs and WRITEs are issued. NRBBUFL is specified in addressable units (bytes for IBM).

NRBDMODE

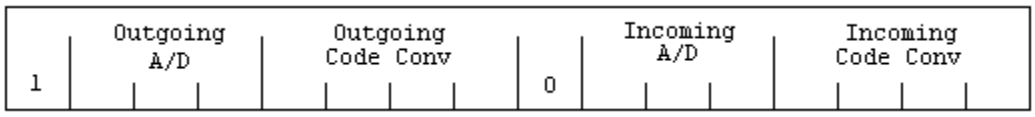
Specify NRBDMODE (datamode) when communicating to a non-IBM host. The transmitting NetEx program specifies NRBDMODE on either an SCONNECT, SCONFIRM, SWRITE, SCLOSE, or SDISCONNECT request. It is always specified as a 16-bit quantity. Datamode is forwarded through all layers of NetEx. When the receiving program receives the data, the datamode specified by the transmitter, with possible modifications as described below, is inserted into the NRB associated with the SREAD or SOFFER request.

Datamode supports two basic modes: manual and auto datamode.

Manual Datamode

Use manual datamode to specify the assembly/disassembly and code conversion functions on both adapter output and adapter input. In manual datamode, the caller has total control over the adapter facilities. The user must determine which assembly/disassembly modes and code conversion tables are meaningful to the two adapters involved in the transfer. Refer to the appropriate Adapter Hardware Reference Manuals for the adapters being used.

Manual datamode has the following format:



1

This number in the high order bit is the manual mode indicator.

Outgoing A/D

Data assembly/disassembly to be performed on data as it goes out onto the network. This information is added to the transmit data function code when the user data goes over the network.

Outgoing Code Conv

Code conversion to be performed on data as it goes out onto the network. This information is added to the transmit data function code when the user data goes over the network.

Incoming A/D

Data assembly/disassembly to be performed on data as it goes from the network to the receiving program. This information is added to the input data function code when the receiving driver gets the message from the network.

Incoming Code Conv

Code conversion to be performed on data as it goes from the network to the receiving program. This information is added to the input data function code when the receiving driver gets the message from the network. Use this field only if the receiving host-processor adapter supports code conversion. When the receiving program receives a block that was sent using a manual datamode, the reading NRB is set with the exact datamode field used to send the block.

Auto Datamode

Use auto datamode for all common NetEx transfers. When auto datamode is selected, the user identifies the source and destination character sets, and NetEx selects the appropriate assembly/disassembly and code conversion. NetEx uses hardware code conversion whenever possible.

Auto datamode supports three conversion options.

Bit Stream

The bit pattern is precisely reproduced in the destination machine.

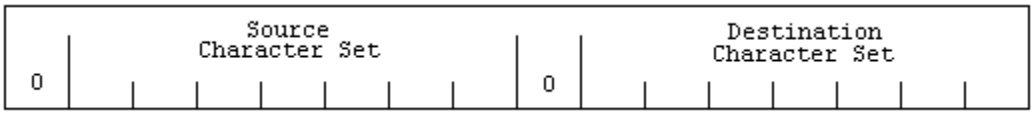
Octet

Eight-bit binary quantities are sent from one machine to another, using an 8-bit byte representation appropriate to each machine. On a Unisys 2200 host, the bottom eight bits of each quarter word are sent or received.

Character

Character information is sent from one machine to another with a full range of character assembly and code conversion options.

The conversion options are selected in the NRBDMODE word. The auto datamode has the following format in the NRBDMODE word:



0

This number in the high order bit is the auto datamode indicator.

Source Character Set

This indicates the conversion option (from Table 2) of the data used in the write buffer of the transmitter.

0

This number in the high bit of the low order byte is reserved.

Destination Character Set

This indicates the conversion option (from Table 2) of the data going to the destination program. For example, a conversion from EBCDIC (3) to ASCII (2) would be entered as the hexadecimal value of 0302 or decimal value of 770.

Table 2. Auto Datamode Character Sets

Indicator	Conversion Option

Indicator	Conversion Option
0	Bit stream mode
1	Octet mode
2	ASCII (8 bit)
3	EBCDIC
4	Reserved
5	BCD (Honeywell)
6	Field-data (UNISYS)
7	Display code (CDC)

The following list describes the processing rules for auto datamode:

- The transmitting NetEx examines the source character set. The character set implicitly specifies the method used to represent those characters on the transmitting machine. NetEx selects an assembly/disassembly mode so that those characters are sent over the network as an 8-bit quantity. If code conversion hardware is installed in the transmitting adapter, NetEx selects the proper hardware code conversion function to convert the character set before the information is sent over the network.
- The receiving NetEx examines the datamode field and selects an A/D mode to convert the 8-bit quantities coming over the network to the bit configuration used by the destination character set. If code conversion hardware is installed in the receiving adapter and code conversion is required, NetEx selects the proper hardware code conversion function.
- If neither adapter has code conversion and the character sets in the datamode field are still not equal, then software code conversion is performed and the two fields are set equal.
- If the destination character set is a 6-bit code, code conversion hardware is required on the 6-bit character machine.

NRBTIME

NRBTIME specifies the length of elapsed time that the associated read-type command remains in effect. If a time interval equal to the number of seconds in NRBTIME has elapsed and no data or connection information has arrived to satisfy the READ or OFFER, then the request ends with an error.

If the value in NRBTIME is 0, then the request waits indefinitely.

NRBCLASS

NRBCLASS specifies the class of service, the protocol used to move data between host processors. The user may select type 1 protocol (available in release 1 and 2 NetEx products) or type 2 protocol (available in release 2 and 3 NetEx products). Type 2 protocol includes block segmenting and alternate path retry; type 1 does not. The following classes may be selected:

0

Use class determined by the Network Configuration Table (NCT). (See “Installation” on page 73 for more information.)

1

Use Version 1 NetEx protocol. This protocol is supported in Release 1 and Release 2 NetEx products, but is not supported in Release 3 NetEx.

2

Use Version 2 NetEx protocol. This protocol is supported in Release 2 and Release 3 NetEx products, but is not supported in Release 1 NetEx. This version of NetEx supports class 2 protocol.

The value specified in this field is restricted by the protocols defined in the NTCROUTE statements. For example, if a route is defined as type 1 protocol only, specifying NRBCLASS=2 results in an error.

Typically, the user should select an NRBCLASS of 0 and let NetEx use the protocol specified by the NTCROUTE statement for that route. If both the NTCROUTE statement and the NRBCLASS are 0, use type 2 protocol.

NRBMAXRT (Deprecated)

NRBMAXRT (maximum rate) is a connection negotiation parameter that specifies the maximum data rate possible for the connection. The NRBMAXRT value is based on the user's specification or the physical characteristics of the links between the two programs calling NetEx. This field is designed for those programs that use variable rate methods of communication. NetEx protocol sends data at a rate that the facilities such as link adapters are not overloaded by sending blocks of data to them faster than they are prepared to handle. If the user specifies a zero in this field, then it is assumed that the user wishes the highest data rate possible (no throttling). During the negotiation process, NetEx examines the speed and propagation delays of the links to determine the maximum data rate that can be used by the link.

The units of this field are expressed as a 16-bit positive quantity giving the link speed in thousands of bits per second. Thus a connection using a terrestrial link adapter whose line speed was generated as 230,000 bits per second has 230 placed in this field.

MAXRT and the throttling concept directly apply only to the transmitting portion of a given connection. The other party in the connection may be working with completely different throttling parameters, and the corresponding program has no direct way of knowing the remote transmitter's data rate parameters.

Upon completion of the OFFER or CONFIRM request, this field has a nonzero value that contains the maximum throughput that is possible to the connection. This is based on the user's original request and the characteristics of the communications link between the two.

NRBBLKI and NRBBLKO

NRBBLKI and NRBBLKO are connection negotiation parameters that specify the maximum amount of data that the calling program expects to read at one time during the coming connection. This parameter should be provided with the CONNECT or OFFER request. During the protocol negotiation process, the NRBBLKI of one program is compared with the NRBBLKO (output maximum buffer size) specified at the other end, and the lesser of the two values are returned in the two respective fields.

For the connecting program, the negotiated results will be returned in the NRB along with the confirm data read following the connect. The offering program will receive the negotiated values on completion of the offer and hence may decide if the negotiated values are acceptable for the work at hand.

The NetEx installation systems programmer must supply the following values controlling these buffer sizes:

1. The default input and output block sizes to be used, if these are not specified (left zero) by the caller.
2. The maximum input and output block sizes permitted by the installation.

As an example of the block negotiation process: Program A issues a connect with NRBBLKI = 256 and NRBBLKO = 4096. The offering program B to which A will connect specifies 64K for both, allowing the connector to set any reasonable value in these fields. When the offer completes, B sees NRBBLKO = 256 and NRBBLKI = 4096, the minimum of the two sets of values. When A's read following the connect completes, it will see NRBBLKI = 256 and NRBBLKO = 4096, which are the same values as B with the directions reversed.

If the connection established is a network or driver level connection, then NRBBLKO and NRBBLKI may be adjusted to reflect the maximum size of data block that may be sent as a datagram on the path specified by the connect. If the application negotiated size is smaller than the maximum NPDU size, then the negotiated parameters will be unchanged. If the maximum NPDU size is smaller, then this maximum size will be returned in both NRBBLKI and NRBBLKO.

Two default options are available with these fields. If a zero is specified in either one of these fields, then the value used for negotiation will be an installation supplied default that is provided at NetEx installation time. If the value in this field is the machine representation of -1, then the size used for negotiation will be the maximum size available for that installation, which is also a parameter specified at initialization time.

Note: The values implied by zero or -1 will be used for negotiation of the connection block sizes. The actual size negotiated will be supplied in these fields on completion of the connect or offer.

For Network layer requests, the NRBBLKI and NRBBLKO fields are used to inform the Network layer of the maximum amounts of Odata and Pdata that will be used to send and receive data in this connection. These lim-its are dependent on the following:

- The buffer capacities generated in both the local and remote copies of NetEx.
- The physical limitations of the media connecting the two hosts.

When this NOFFER completes with a Connect Indication, then these fields will have the actual limits for Odata and Pdata size in the connection sent to them. Unlike other layers of NetEx service, the Network Service will return the maximum that is available if the caller's size request is not available. The caller must scale its buffer sizes downward accordingly.

The maximum size of Pdata is specified in addressable units. The maximum amount of Odata is specified in octets.

NRBPROTA and NRBPROTL

The NRBPROTA and NRBPROTL are connection negotiation parameters that permit the application to provide Odata to the called layer of NetEx. NRBPROTA specifies the address of the buffer containing the Odata, and NRBPROTL specifies the number of octets of Odata in that buffer.

When a write-type command is issued, the Odata provided (if any) will be added to the message, and eventually delivered as Odata to the receiving application's read-type command. As a result, this is a second buffer that is handled in a similar way to the Pdata that is specified by NRBBUFA and NRBLEN/NRBUBIT. There are some distinct differences that are as follows:

- Odata is always sent and received in "octet mode," which means it will be represented in the best way that the particular host can handle strings of 8-bit binary quantities (for example, 1/byte, 4/36-bit word, and so on).
- The maximum amount of Odata that may be sent is limited. This maximum is installation dependent and may typically be 256 bytes or less. Each version of NetEx will have a generated maximum on the number of bytes of Odata that it is prepared to accept in incoming messages. In the Network,

Transport, and Session levels, the maximum amount of Odata that may be sent or received will be the minimum of the Odata sizes generated on each host.

Users should be warned that sending excessive amounts of Odata with normal transmissions may result in a "fissioning" of network messages, which increases network traffic and decreases network performance, often by a factor of two.

Note: Not all implementations of NetEx support the use of Odata. Consult Network Executive Software personnel before using Odata to determine whether it is available.

On a write-type operation, no Odata will be sent if NRBPROTL is zero. If a non-zero length is specified, then the Odata will be transmitted along with the Pdata, if present. When the read takes place, the Odata will be placed in the address specified by NRBPROTA and its incoming length will be set in NRBPROTL.

NRBPROTL always contains the length of the Odata in octets, not "addressable units."

The protocol field has special significance when used with Driver level requests, in that the Odata contains the network Message Proper, where the Pdata contains the Associated Data.

NRBRESV1 and NRBRESV2

NRBRESV1 and NRBRESV2 are reserved for future NetEx enhancements. Applications should leave binary zeroes in these fields.

NRBPNAME

NRBPNAME is an eight-byte field used for both SCONNECT and SOFFER requests to specify the OFFERed name, the name of the process to be matched when the OFFER and CONNECT requests meet. Names of all processes are uppercase alphanumeric data up to eight characters long. Names less than eight characters long are padded with blanks (left justified blank filled). Process names are converted to the ASCII character set for transmission between hosts, so only those characters that are uppercase alphanumeric should be used during the name matching process. After the connection sequence completes, the offered (SOFFER) process contains the unique identifier (for example, jobname, logon-id, or userid) of the connecting process in the NRBPNAME field.

NRBHNAME

NRBHNAME is an eight-byte field used by the SCONNECT service to specify the symbolic name of the host computer that is addressed to match an OFFER request. Names of all hosts are specified by the installation systems programmer. All host names are uppercase alphanumeric data that are up to eight characters long. Names less than eight characters long should be padded with blanks (left justified blank filled).

NRBRESV3 and NRBRESV4

NRBRESV3 and NRBRESV4 are reserved for possible future NetEx enhancements. Applications should leave binary zeroes in these fields.

NRBOSDEP

NRBOSDEP is reserved for internal use. NetEx software uses this field to service and monitor the progress of NRB requests. The contents of these fields is maintained by NetEx during the course of a session.

If these fields are altered by the calling program, the results are unpredictable.

Creating an NRB

A single NRB should be created before an calling program OFFERs or CONNECTs to another program. The NRB is 40 words long and should initially be zero-filled, or if programming in Assembler, may be assigned initial values using the NRB macro. Initially, programs may create several NRBs. If programming in Assembler, align the NRB on a fullword boundary.

If several NRBs are required to service a single connection, they should be duplicated from the initial NRB, as described in the following paragraphs.

Duplicating an NRB

Duplicating NRBs is necessary when using multiple NRBs to service a single connection. By duplicating the NRB, the connection-negotiation parameters, the connection reference number, and the internal NRBOSDEP information is preserved, allowing the duplicate NRB to be valid.

To duplicate an NRB, wait until the initial CONNECT or OFFER has completed successfully. Then copy the entire working NRB through the NRBOSDEP field to a blank NRB at a different location. The second NRB can now be used for NetEx requests.

Assembler Interface

The NetEx MASM DEF element NXDEFS contains a set of simple procedures designed to facilitate NetEx calls from assembler programs. These procedures and associated definitions can be made available to MASM by means of a $INCLUDE “NXDEFS” statement in the program, and placing the NXDEFS omnibus element in an appropriate file in MASM’s search hierarchy (ASM$PF, SI, SO, RO).

The rest of this section describes the Assembler NRBs, the general Procedure format, and each NetEx assembler call.

Assembler NetEx Request Blocks

The MASM user builds an NRB by reserving a 40-word data area. Various members of this area will hold the information to be transferred to NetEx, and others will contain the information that is returned by NetEx. If more than one NRB will be required to service a program, then several of these NRB areas must be reserved.

The NRB word definitions shown below are available to the assembler programmer in the MASM omnibus element NXDEFS. In order to use these names, a $INCLUDE “NXDEFS” statement must appear in the assembly code.

Table 3. Assembler NRB

Element	Name	Function
NRB+0	NRBSTAT	NetEx request status returned to user
NRB+1	NRBIND	Data type indication from OFFER/READ
NRB+2	NRBLEN	Length of data
NRB+3	NRBUBIT	Unused bit count
NRB+4	NRBREQ	User request code
NRB+5	NRBNREF	NetEx ref number identifying connection
NRB+6	NRBBUFA	Starting address of user data
NRB+7	NRBBUFL	Length of user's buffer
NRB+8	NRBMODE	Datamode for write-type request
NRB+9	NRBTIME	Read request timeout in seconds
NRB+10	NRBCLASS	Class of service
NRB+11	NRBMAXRT	Maximum data rate permitted
NRB+12	NRBBLKI	Max buffer size for input requests
NRB+13	NRBBLKO	Max buffer size for output requests
NRB+14	NRBPROTA	Odata address
NRB+15	NRBPROTL	Odata length in octets

Table 3. Assembler NRB		
Element	**Name**	**Function**
NRB+16	NRBRESV1	Reserved
NRB+17	NRBRESV2	Reserved
NRB+18	NRBPNAME	Name of process to OFFER/CONNECT (8 bytes)
NRB+28	NRBHNAME	Name of destination host (8 bytes)
NRB+22	NRBRESV3	Reserved
NRB+23	NRBUSER	Installation use
NRB+24 to NRB+39	NRBOSDEP	Reserved for op system dependent information

Assembler Procedure Format

All of the NetEx assembler procedures have the following format,

Call	Parameters
`nxcall[W]`	`nrb [err-return]`

nxcall

This is the procedure name. The name may be one of the following: SOFFER, SCONNECT, SCONFIRM, SWRITE, SREAD, SCLOSE, or SDISC. More information about each of these NetEx procedures is presented later in this section.

nxcallw

If this optional parameter is specified, return will not be made to the caller until the request has completed (NRBSTAT zero or positive). If not specified, return will be made immediately after preliminary error-checking of the request (by NetEx).

nrb

This is the address of the NetEx request block (NRB) that is to be used with this procedure. All relevant fields must have been previously set or cleared in the NRB by the caller. NRBSTAT and NRBIND are always cleared by the procedure.

This field may have the following form:

```
U-field[,X-reg[,J-desig]]
```

err-return

If specified, this is optional parameter is the address to which control will be transferred if the request completes with a nonzero return code (indicating an error). Note that this does not apply to the immediate return with an in-progress status (negative value).

Register Usage

All calls use and destroy the minor register set. A0 contains the NRB address on return (except for SWAIT, where it is undefined).

Assembler Command Examples

The following statements are sample NetEx calls:

```
L,U             X9,NRB1
SOFFERW  X9

SWRITE   NRBW,,U     WRITERR

SREADW   RDNRBA,,U
```

SOFFER Assembler Call

The SOFFER call provides a means for a program desiring to accept a connection from a caller on the network to post the availability of the service.

Prior to issuing an SOFFER call, the user must provide an NRB (described in "NetEx Request Block") to be used by the user interface. All NRB fields which are not defaults or have not been assembled into the NRB data area(s) must be filled in by the program before calling NetEx. The areas used by the SOFFER call include the following:

NRBPNAME Name of process to offer.

NRBBLKI Maximum buffer size for input requests (may be changed during the NetEx negotiation process).

NRBBLKO Maximum buffer size for output requests (may be changed during the NetEx negotiation process).

NRBTIME Number of seconds before the request times-out.

NRBBUFA Starting address to receive data coining back with the other programs CONNECT re-quest.

NRBBUFL Length of buffer to receive data coming back with the other programs CONNECT re-quest.

NRBMAXRT Maximum data transmission rate permitted in thousands of bits per second (may be changed during the NetEx negotiation process).

Call	Parameters
`SOFFER[W]`	`nrb [err-return]`

SOFFER

This is the procedure name.

SOFFERW

If this optional call is specified, the return will not be made to the caller until the request has completed. If not specified, return will be made immediately after preliminary error-checking of the request (by NetEx).

nrb

This is the address of the updated NetEx request block that is to be used with this procedure.

err-return

If this optional parameter is specified, this field is the address to which control will be transferred if the request completes with a nonzero return code (indicating an error). Note that this does not apply to the immediate return with in-progress status (negative value).

SCONNECT Assembler Call

The SCONNECT call provides a means for a program to request a session with a program that has previously issued an SOFFER.

Prior to issuing an SCONNECT call, the user must provide an NRB (described in "NetEx Request Block") to be used by the user interface. All NRB fields which are not defaults or have not been assembled into the NRB data area(s) must be filled in by the program before calling NetEx. The areas used by the SCONNECT call include the following:

NRBPNAME Name of process requesting the connection.

NRBHNAME Name of destination host to be connected to.

NRBBLKI Maximum buffer size for input requests (may be changed during the NetEx negotiation process).

NRBBLKO Maximum buffer size for output requests (may be changed during the NetEx negotiation process).

NRBBUFA Starting address of (optional) data to be transmitted to the other program with the SCONNECT request.

NRBLEN Length of (optional) data to be transmitted with the SCONNECT request. Note that this must not exceed the maximum segment size of either host.

NRBUBIT Unused bit count for (optional) data to be transmitted with the SCONNECT request.

NRBMODE Datamode for making data intelligible for users on both ends

NRBMAXRT Maximum data transmission rate permitted in thousands of bits per second (may be changed during the NetEx negotiation process).

Call	Parameters
`SCONNECT[W]`	`nrb [err-return]`

SCONNECT

This is the procedure name.

SCONNTW

If this optional call is specified, the return will not be made to the caller until the request has completed. If not specified, return will be made immediately after preliminary error-checking of the request (by NetEx).

nrb

This is the address of the updated NetEx request block that is to be used with this procedure.

err-return

If this optional parameter is specified, this field is the address to which control will be transferred if the request completes with a nonzero return code (indicating an error). Note that this does not apply to the immediate return with an in-progress status (negative value).

SCONFIRM Assembler Call

The SCONFIRM call provides a means for an offering program to confirm (to the connector) that the connection has been made and the block size, class, and any optional data are acceptable to the offerer. A negative response would be in the form of an SDISC (disconnect).

Before issuing the SCONFIRM, an SOFFER must have completed successfully with a Connect Indication. The user must provide a NRB containing the information supplied by the previous SOFFER.

Also prior to issuing an SCONFIRM call, the user must provide an NRB (described in "NetEx Request Block") to be used by the user interface. All NRB fields which are not defaults or have not been assembled into the NRB data area(s) must be filled in by the program before calling NetEx. The areas used by the SCONFIRM call include the following

NRBBUFA Starting address of (optional) data to be transmitted to the other program with the SCONFIRM request.

NRBLEN Length of (optional) data to be transmitted with the SCONFIRM request. Note that this must not exceed the maximum segment size of either host.

NRBUBIT Unused bit count for (optional) data to be transmitted with the SCONFIRM request.

NRBMODE Datamode for making data intelligible for users on both ends.

Call	Parameters
`SCONFIRM[W]`	`nrb [err-return]`

SCONFIRM

This is the procedure name.

SCONFIRMW

If this optional call is specified, the return will not be made to the caller until the request has completed. If not specified, return will be made immediately after preliminary error-checking of the request (by NetEx).

nrb

This is the address of the updated NetEx request block that is to be used with this procedure.

err-return (optional)

If specified, this field is the address to which control will be transferred if the request completes with a nonzero return code (indicating an error). Note that this does not apply to the immediate return with an in-progress status (negative value).

SREAD Assembler Call

The SREAD call provides a means for a program to receive data from another host or an indication from NetEx of an abnormal condition with the connection.

Before an SREAD can be issued, a connection must be established. The NRB specified must have been used for a previous NetEx request for the particular connection, or a copy of another NRB that has been used to service the desired connection. Defaults for unspecified parameters are the parameters existing in the specified NRB.

Also prior to issuing an SREAD call, the user must provide an NRB (described in "NetEx Request Block") to be used by the user interface. All NRB fields must be filled in by the program before calling NetEx, unless previously used values are to be used again. The areas used by the SREAD call include the following.

NRBBUFA Starting address of data area to be used for receiving data from the other program.

NRBBUFL Length of user buffer to be used to receive data from the other program.

NRBTIME Number of seconds before this request times-out.

Call	Parameters
`SREAD[W]`	`nrb [err-return]`

SREAD

This is the procedure name.

SREADW

If this optional call is specified, the return will not be made to the caller until the request has completed. If not specified, return will be made immediately after preliminary error-checking of the request (by NetEx).

nrb

This is the address of the updated NetEx request block that is to be used with this procedure.

err-return

If this optional parameter is specified, this field is the address to which control will be transferred if the request completes with a nonzero return code (indicating an error). Note that this does not apply to the immediate return with an in-progress status (negative value).

SWRITE Assembler Call

The SWRITE call provides a means for a program to transmit data to another calling program. Before an SWRITE can be issued, a connection must be established. The NRB used to issue the SWRITE must have been used by a previous request that serviced the desired connection, or it must be a copy of another NRB that has serviced the connection. Defaults for unspecified parameters are the existing parameters in the specified NRB.

Also prior to issuing an SWRITE call, the user must provide an NRB (described in "NetEx Request Block") to be used by the user interface. All NRB fields which are not defaults or have not been assembled into the NRB data area(s) must be filled in by the program before calling NetEx. The areas used by the SWRITE call include the following:

NRBBUFA Starting address of data to be transmitted to the other program with the SWRITE request.

NRBLEN Length of data to be transmitted with the SWRITE request.

NRBUBIT Unused bit count for data to be transmitted with the SWRITE request.

NRBMODE Datamode for making data intelligible for users on both ends.

Call	Parameters
`SWRITE[W]`	`nrb [err-return]`

SWRITE

This is the procedure name.

SWRITEW

If this optional call is specified, the return will not be made to the caller until the request has completed. If not specified, return will be made immediately after preliminary error-checking of the request (by NetEx).

nrb

This is the address of the updated NetEx request block that is to be used with this procedure.

err-return

If this optional parameter is specified, this field is the address to which control will be transferred if the request completes with a nonzero return code (indicating an error). Note that this does not apply to the immediate return with in-progress status (negative value).

SCLOSE Assembler Call

The SCLOSE call provides a means for a program to transmit data to another calling program, and indicate that this is the last data to be sent.

Before an SCLOSE can be issued, a connection must be established. The NRB used to issue the SCLOSE must have been used by a previous request that serviced the desired connection, or it must be a copy of another NRB that has serviced the connection. Defaults for unspecified parameters are the existing parameters in the specified NRB.

After a program has issued an SCLOSE, no other data may be written by that program. If the other program had previously issued an SCLOSE, the data is written and then the connection is gracefully disconnected. If the other program has not issued an SCLOSE, it is still free to write data to the program that did issue the SCLOSE.

Also prior to issuing an SCLOSE call, the user must provide an NRB (described in "NetEx Request Block") to be used by the user interface. All NRB fields which are not defaults or have not been assembled into the NRB data area(s) must be filled in by the program before calling NetEx. The areas used by the SCLOSE call include the following.

NRBBUFA Starting address of data to be transmitted to the other program with the SCLOSE request.

NRBLEN Length of data to be transmitted with the SCLOSE request.

NRBUBIT Unused bit count for data to be transmitted with the SCLOSE request.

NRBMODE Datamode for making data intelligible for users on both ends.

Call	Parameters
`SCLOSE[W]`	`nrb [err-return]`

SCLOSE

This is the procedure name.

SCLOSEW

If this optional call is specified, the return will not be made to the caller until the request has completed. If not specified, return will be made immediately after preliminary error-checking of the request (by NetEx).

nrb

This is the address of the updated NetEx request block that is to be used with this procedure.

err-return

If this optional parameter is specified, this field is the address to which control will be transferred if the request completes with a nonzero return code (indicating an error). Note that this does not apply to the immediate return with an in-progress status (negative value).

SDISC Assembler Call

The SDISC (disconnect) call provides the means for any connected program to terminate a session. The request is immediate and any data currently in transport may not be delivered. If data delivery is required, it is the responsibility of the session requesting the disconnect to wait for confirmation of previous SREAD or SWRITE commands by the corresponding program before issuing the SDISC call.

Before an SDISC can be issued, the user must provide an NRB that has been previously used to service the desired connection, or a copy of an NRB used to service the connection.

Also prior to issuing an SDISC call, the user must provide an NRB (described in "NetEx Request Block") to be used by the user interface. All NRB fields which are not defaults or have not been assembled into the NRB data area(s) must be filled in by the program before calling NetEx. The areas used by the SDISC call include the following.

NRBBUFA Starting address of (optional) data to be transmitted to the other program with the SDISC request.

NRBLEN Length of (optional) data to be transmitted with the SDISC request. Note that this must not exceed the maximum segment size of either host.

NRBUBIT Unused bit count for (optional) data to be transmitted with the SDISC request.

NRBMODE Datamode for making data intelligible for users on both ends.

Call	Parameters
`SDISC[W]`	`nrb [err-return]`

SDISC

This is the procedure name.

SDISCW

If this optional parameter is specified, return will not be made to the caller until the request has completed. If not specified, return will be made immediately after preliminary error-checking of the request (by NetEx).

nrb

This is the address of the updated NetEx request block that is to be used with this procedure.

err-return

If this optional parameter is specified, this field is the address to which control will be transferred if the request completes with a nonzero return code (indicating an error). Note that this does not apply to the immediate return with an in-progress status (negative value).

NetEx Procedure Code

The code generated by the NetEx assembler procedures is as follows.

```
L       A0,nrb-address          . request block
L       A5,req-type             . type of request
S       A5,NRBREQ,A0            . request code
SZ      NRBRC,A0                . clear return code
SZ      NRBIND,A0               . clear indicator
L       A4,(NXIFLV,NTX$DB0)     . interface level and 0-bank
LXI,U         Xll,NTX$COMN+l*/15      . into utility I-bank
LBJ           Xll,01000               . call NETEX
```

If err-return was specified, the following results:

```
TP      nrbrc,a0                . request in progress?
J       $+3                     . yes
TZ      NRBRC,A0                . error code?
J       err-return              . yes
;
```

Note that since the user's utility I-bank and D-bank are debased by the NetEx call, the NRB and data buffer may not reside in these banks. NetEx must have these data areas "visible" in the user interface.

SWAIT Assembler Call

The SWAIT procedure provides a means to wait for current NetEx operations to finish. Unlike the higher level language interfaces, this interface does not accept NRB's to check. Instead, it is the responsibility of the assembly language user to check the status of the NRB's when control is returned.

When SWAIT is called, it checks if any operations have completed since the last time control was returned from an SWAIT call. If an operation completed, control is immediately returned to the caller. If no operations have completed, the user is suspended until an operation completes.

Because SWAIT "remembers" if any operation has completed since the last call, SWAIT may return with no apparent NRB completions. This occurs if the completion occurred after the user regained control and before the NRB's were checked. The SWAIT code flags the completion and then the user processes the NRB. On the next SWAIT call, the completion flag is set and control is returned, but no new NRB's have completed.

The only current error which can be reported from an SWAIT is that NetEx is not currently running. In this case, the user cannot be suspended and control is returned with the error status. If the user wishes to wait until NetEx is up, the user should wait and try again until SWAIT returns with a good status. It is possible for the user to be aborted with a common bank reload error when NetEx starts up. This is a result of a call (SWAIT, SCONNECT, or SOFFER) reaching the common bank just as the reload occurs.

There is only one WAIT request. It waits for completions at any call level. If the user is using SESSION and DRIVER requests, the SWAIT will wait for any SESSION request or any DRIVER request. For coding convenience the names SWAIT, TWAIT, NWAIT, and DWAIT are defined.

Call	Parameters
`SWAIT`	`err-return`

SWAIT
This is the procedure name.

err-return
This required parameter is the address to transfer control to if the request completes with an error indication. The error status which normally would be returned in NRBSTAT is returned in A0. Currently, the only error is "NetEx is not running".

Multiple Activities

NetEx is designed to allow multiple user activities. It is common to have one activity to process all output requests and one for all input requests. There is no limit imposed by NetEx on the number of activities.

As with the other interfaces, SWAIT allows multiple activities. There is an attempt to process requests in the order that they were received, but it is common to have the last caller and the first call active if a completion occurs just as the last caller calls. The assembly language interface and the higher level language interfaces are not compatible. The higher level language interfaces need to receive control on each completion. If both the assembly and higher level interfaces are to be used, only one activity should be used. That activity should use the SWAIT(-l) type interface and call the user's assembly language completion routine for the assembler completions.

FORTRAN Interface

NetEx includes a library of subroutines that are designed to be called by FORTRAN high level language programs. When the user makes a call to the user interface, he will supply the appropriate information, in parameter format, to pass to NetEx. An interface module resides in the NetEx release file. The module name is nxiftn.

FORTRAN programs written to use NetEx may be used on other machines that use NetEx, provided that changes are made to the program to account for different word sizes, etc. One such change is the data length used when reading and writing data. Data lengths are specified to NetEx as a number of addressable units. An addressable unit is the amount of information obtainable from a distinct memory location on a particular machine. For UNIVAC, an addressable unit is one 36-bit word.

There are two components that are used to establish working communications through NetEx: one or more NetEx Request Blocks (NRBs) that must be supplied by the FORTRAN caller, and the NetEx-provided subroutines that are used to invoke NetEx services. The NRB is described first, followed by the calls to the subroutines. The calls are presented in the following order (the approximate order in which they are used).

- SCONN
- SCONF
- SREAD
- SWRIT
- SCLOSE
- SWAIT
- SDISC

The use of these calls is described in “Intertask Communication” on page 9 of this manual. A FORTRAN example of a program using NetEx follows the macro description in this section. The formats of the calls are presented using the conventions stated in the preface of this manual.

FORTRAN NetEx Request Blocks

The FORTRAN user builds an NRB by declaring it to be an array of 40 INTEGER elements. Various members of this array will hold the information to be transferred to NetEx, and others will contain the information that is returned by NetEx. If more than one NRB will be required to service a program, then several of these NRB arrays must be declared. Before these NRB arrays are used for any NetEx request, it is advisable to zero all the elements of the array. This will allow defaults for fields not explicitly used by the caller to take effect. Thus a sample declaration might be as follows.

```
INTEGER RNRB(40), WNRB(40)
DATA    RNRB /40*0/
DATA    WNRB /40*0/
```

The NetEx FORTRAN subroutines have the philosophy that arguments commonly passed to NetEx (such as a data buffer address) will be passed as parameters to the subroutine. “Exotic” parameters to be passed to NetEx, such as maximum input block size, will be supplied by storing the desired value in the proper member of the NRB array. When the request completes, the FORTRAN program directly accesses the

desired elements of the NRB array to determine if the operation completed properly. If NRB is declared as a 40 element array of integers, the elements of the array will be as shown in the following table.

Table 4. FORTRAN NRB

Element	Type	Name	Function
NRB(l)	INTEGER	NRBSTAT	NetEx request status returned to user
NRB(2)	INTEGER	NRBIND	Data type indication from OFFER/READ
NRB(3)	INTEGER	WRELEN	Length of data
NRB(4)	INTEGER	NRBUBIT	Unused bit count
NRB(5)	INTEGER	NRBREQ	User request code
NRB(5)	INTEGER	NRBNREF	NetEx ref no identifying connection
NRB(7)	INTEGER	NRBBUFA	Starting address of user data
NRB(8)	INTEGER	NRBBUFA	Length of users buffer
NRB(9)	INTEGER	NRBMODE	Datamode for write-type request
NRB(10)	INTEGER	NRBTIME	Request timeout in seconds
NRB(11)	INTEGER	NRBCLASS	Class of service
NRB(12)	INTEGER	NRBMAXRT	Maximum data rate permitted
NRB(13)	INTEGER	NRBBLKI	Max buffer size for input requests
NRB(14)	INTEGER	NRBBLKO	Max buffer size for output requests
NRB(15)	INTEGER	NRBRESV1	Odata address
NRB(16)	INTEGER	NRBRESV2	Odata length (octets)
NRB(17)	INTEGER	NRBPROTA	Reserved
NRB(18)	INTEGER	NRBPROTL	Reserved
NRB(19)	CHARACTER*8	NRBPNAME	Name of process to OFFER/CONNECT (8b)
NRB(21)	CHARACTER*8	NRBHNANE	Name of destination host (8 bytes)
NRB(23)	INTEGER	NRBRESV3	Reserved
NRB(24)	INTEGER	NRBUSER	Installation use
NRB(25) to NRB(40)		NRBOSDEP	Reserved for op system dependent info

SOFFR FORTRAN Call

The SOFFR (offer) subroutine provides a means for a program desiring to accept a connection from a caller on the network to post the availability of the service. Before issuing an SOFFR call, the user must provide an NRB (described in "NetEx Request Block") to be used by the user interface. Also, NetEx must be active in the system.

Call	Operation	Parameters
CALL	SOFFR[W]	(nrb, buffer, length, timeou, pname)

The following parameters were shown in the SOFFR call format. The parameters must be specified in the order presented. If parameters are omitted, the commas must still be used.

SOFFR
This is the verb for the call. Either SOFFR or SOFFRW must be specified.

SOFFRW
This specifies that the calling program must wait for the call to complete before processing is resumed.

nrb
This required parameter is the address of the NRB data area that is to be passed to NetEx.

buffer
This optional parameter specifies the start of an array that is to receive data sent by the corresponding SCONNECT request.

length
This optional parameter specifies is the length of the buffer (in addressable units - words for Unisys) that will hold the data sent by the corresponding SCONNECT. When called, length should contain the maximum size of the buffer. On return the NRBLEN (NRB(3)) field will contain the number of addressable units of information actually sent to the offering application. The data type of length should be INTEGER.

timeou
This optional parameter indicates the number of seconds that the OFFER request should remain outstanding. If no application connects during this interval, then the OFFER will end abnormally. If timeou is specified as zero, the OFFER will remain outstanding indefinitely.

pname
This optional parameter is the alphabetic name of the process to be offered to the corresponding calling program. The name offered is arbitrary, but must be known to the connecting program. This name may be provided as a CHARACTER*8 variable or as a string in the CALL statement if padded with blanks to eight characters in length.

SOFFR Entry Parameters

The following NRB fields are used by SOFFR on entry.

NRBBUFA Address for incoming Pdata

NRBBUFL Length of buffer to hold Pdata.

NRBTIME Number of seconds offer outstanding.

NRBBLKO Maximum transmission size acceptable.

NRBBLKI Maximum reception size acceptable.

NRBMAXRT Limit on transmission speed.

NRBPROTA Address for incoming Odata.

NRBPROTL Length of buffer to hold Odata.

NRBPNAME Application name to offer.

SOFFR Results

The following NRB fields are updated when SOFFR completes.

NRBSTAT Success/failure code.

NRBIND Contains Connect Indication.

NRBLEN Length of incoming Pdata.

NRBUBIT Unused bit count of Pdata.

NRBNREF Sref assigned this connection.

NRBBLKO Maximum transmission Pdata size.

NRBBLKI Maximum reception Pdata size.

NRBMAXRT Max transmission speed of path.

NRBPROTL Length of Odata received.

NRBHNAME Name of host where S-conn originated.

NRBPNAME Name of application where S-conn originated.

SCONN FORTRAN Call

The SCONN (connect) call provides a means for a program to request a session with a program that has issued an SOFFR. Before issuing the SCONN, an NRB must be provided for use by the user interface.

Call	Operation	Parameters
CALL	SCONN[W]	(nrb, buffer, length, datamd, pname, hname)

The following parameters were shown in the SCONN call format. The parameters must be specified in the order presented. If parameters are omitted, the commas must still be used.

CALL

This is the required standard high level call instruction.

SCONN

This is the verb for the call. Either SCONN or SCONNW must be specified.

SCONNW

This indicates that the calling program should wait for the call to complete before processing is resumed.

nrb

This required parameter is the address of the NRB data area that is to be passed to NetEx.

buffer

This optional parameter specifies the start of an array that holds the user data that is to be sent to the corresponding application.

length

This optional parameter specifies the length of the data (in addressable units - words for Unisys) that is to be sent to the corresponding application, to be presented with the completion of the corresponding application's OFFER request. If no data needs to be sent to the other application, the length may be set to zero. The data type of length should be INTEGER.

datamd

This optional parameter specifies the datamode that is to be used to send the connect data to the corresponding application. The data type of datamd should be INTEGER. Refer to NRBDMODE in "NetEx Request Block" on page 25 for a discussion of the datamode parameter.

pname

This optional parameter specifies the alphabetic name of the process SOFFR'd by the corresponding calling program. The name offered is determined by the other calling program. This name may be provided as a CHARACTER*8 variable or as a string in the CALL statement if padded with blanks to eight characters in length.

hname

This required parameter specifies the alphabetic name of the host computer to be accessed to determine if the correct SOFFR is available. The "names" of the host computers in the network are determined by the NetEx installation systems programmer. This may be provided as a

CHARACTER*8 variable or provided as a string in the CALL statement if padded with blanks to eight characters in length.

SCONN Entry Parameters

The following NRB fields are used by SCONN on entry.

NRBBUFA	Address of outgoing Pdata.
NRBLEN	Length of outgoing Pdata.
NRBUBIT	Pdata unused bit count.
NRBDMODE	Datamode of Pdata.
NRBBLKO	Maximum transmission size acceptable.
NRBBLKI	Maximum reception size acceptable.
NRBMAXRT	Limit on transmission speed.
NRLBPROTL	Address for outgoing Odata.
NRBPROTA	Length of outgoing Odata.
NRBHNAME	Alphanumeric “host” name.
NRBPNAME	Alphanumeric “application” name.

SCONN Results

The following NRB fields are updated when SCONN completes.

NRBSTAT	Success/failure code.
NRBNREF	Sref (Session ID) assigned.

SCONF FORTRAN Call

The SCONF (confirm) call provides a means for an offering program to confirm (to the connector) that the connection has been successfully completed. A negative response to an SCONN would be an SDISC. Before issuing the SCONF call, an SOFFR must have completed successfully by receiving an SCONN response. The calling program must provide a NRB with an NRBNREF relating to this session.

Call	Operation	Parameters

`CALL`	`SCONF[W]`	`(nrb,`
		`buffer,`
		`length,`
		`datamd)`

The following parameters were shown in the SCONF call format. The parameters must be specified in the order presented. If parameters are omitted, the commas must still be used.

CALL

This is a required standard high level call instruction.

SCONF

This is the verb for this call. Either SCONF or SCONFW must be specified. SCONFW specifies that the calling program must wait for the call to complete before processing is resumed.

nrb

This required parameter is the address of the NRB data area that is to be passed to NetEx.

buffer

This optional parameter specifies the start of an array that holds the user data that is to be sent to the corresponding application.

length

This optional parameter specifies the length of the data (in addressable units – Unisys words) that is to be sent to the corresponding application, to be presented with the completion of the corresponding application's SREAD request. If no data needs to be sent to the other application, the length may be set to zero. The data type of length should be INTEGER.

datamd

This optional parameter specifies the datamode to be used to send the connect data to the corresponding application. The data type of datamd should be INTEGER. Refer to NRBDMODE in "NetEx Request Block" for a discussion of the datamode parameter.

SCONF Entry Parameters

The following NRB fields are used by SCONF on entry:

NRBBUFA Address of outgoing Pdata (move mode).

NRBLEN Length of outgoing Pdata.

NRBUBIT Pdata unused bit count.

NRBDMODE Datamode of Pdata.

NRBPROTA Address of outgoing Odata.

NRBPROTL Length of outgoing Odata.

SCONF Results

The following NRB fields are updated when SCONF completes:

NRBSTAT Success/failure code.

SREAD FORTRAN Call

The SREAD request provides a means for a program to receive data from another host. NetEx assumes that the calling program that wrote the data specified code conversion which makes it readable for the receiver. Before an SREAD can be issued, a connection must be established.

Call	Operation	Parameters
CALL	SREAD[W]	(nrb, buffer, length, timeou)

The following parameters were shown in the SREAD call format. The parameters must be specified in the order presented. If parameters are omitted, the commas must still be used.

CALL
This is the required standard high level call instruction.

SREAD
This is the verb for this call. Either SREAD or SREADW must be specified.

SREADW
This parameter specifies that the calling program must wait for the call to complete before processing is resumed.

nrb
This required parameter is the address of the NRB data area that is to be passed to NetEx.

buffer
This required parameter specifies the start of an array that is to receive the data sent by the corresponding application's SWRITE or SCONFIRM request.

length
This required parameter specifies the length of the buffer (in addressable units - words for Unisys) that is to hold the data sent by the corresponding SWRITE. When called, length should contain the maximum size of the buffer. On return, the actual length input will be in NRBLEN (NRB(3)). Programs that wish to work with the Unused Bit Count on input should examine the NRBUBIT field (NRB(4)). The data type of length should be INTEGER.

timeou
This optional parameter specifies the number of seconds that the READ request should remain outstanding. If the corresponding application does not send data during this interval, then the READ will end abnormally. If timeou is specified as zero, then the READ will remain outstanding indefinitely. The programmer should take alternate path retry into consideration when specifying the timeout value. Refer to "Overview".

SREAD Entry Parameters

The following NRB fields are used by SREAD on entry.

NRBBUFA Address for incoming Pdata (move mode).

NRBBUFL Length of buffer to hold Pdata.

NRBTIME Number of seconds offer outstanding.

NRBPROTA Address for incoming Odata.

NRBPROTL Length of buffer to hold Odata.

SREAD Results

The following NRB fields are updated when SREAD completes:

NRBSTAT Success/failure code.

NRBIND Contains data type indication.

NRBLEN Length of incoming Pdata.

NRBUBIT Unused bit count of Pdata.

NRBBLKO Maximum transmission Pdata size.

NRBBLKI Maximum reception Pdata size.

NRBMAXRT Maximum transmission speed of the path.

If a CONFIRM was read, the following results:

NRBHNAME Host connected to.

NRBPNAME Remote user name (RUNID).

SWRIT FORTRAN Call

The SWRIT (write) call provides a means for an application to transmit data to another calling program. Before an SWRIT can be issued, a connection must be established.

Call	Operation	Parameters

`CALL`	`SWRIT[W]`	`(nrb,`
		`buffer,`
		`length,`
		`datamd)`

The following parameters were shown in the SWRIT call format. The parameters must be specified in the order presented. If parameters are omitted, the commas must still be used.

CALL
This is a required standard high level call instruction.

SWRIT
This is the verb for this call. Either SWRIT or SWRITW must be specified.

SWRITW
This specifies that the calling program must wait for the call to complete before processing is resumed.

nrb
This required parameter specifies the address of the NRB data area that is to be passed to NetEx.

buffer
This required parameter specifies the start of an array that holds the user data that is to be sent to the corresponding application.

length
This required parameter specifies the length of the data (in addressable units - words for Unisys) that is to be sent to the corresponding application, to be presented with the completion of the corresponding application's OFFER request. If no data needs to be sent to the other application, the length may be set to zero. The data type of length should be INTEGER.

datamd
This optional parameter specifies the datamode that is to be used to send the connect data to the corresponding application. The data type of datamd should be INTEGER. Refer to NRBDMODE in "NetEx Request Block" for a discussion of the datamode parameter.

SWRIT Entry Parameters

The following NRB fields are used by SWRIT on entry:

NRBBUFA Address of outgoing Pdata.

NRBLEN Length of outgoing Pdata.

NRBUBIT Pdata unused bit count.

NRBDMODE Datamode of Pdata.

NRBPROTA Address of outgoing Odata.

NRBPROTL Length of outgoing Odata.

SWRIT Results

The following NRB fields are updated when SWRIT completes:

NRBSTAT Success/failure code.

SCLOS FORTRAN Call

The SCLOS (close) call provides a means for a calling program to transmit data to another calling program and to indicate that this is the last data to be sent. If the other program has already issued an SCLOS, the session is gracefully disconnected. Before an SCLOS can be issued, a connection must be established.

Call	Operation	Parameters

`CALL`	`SCLOS[W]`	`(nrb,`
		`buffer,`
		`length,`
		`datamd)`

The following parameters were shown in the SCLOS call format. The parameters must be specified in the order presented. All parameters are required.

CALL
This is a standard high level call instruction.

SCLOS
This is the verb for this call. Either SCLOS or SCLOSW must be specified.

SCLOSW
This parameter specifies that the calling program must wait for the call to complete before processing is resumed.

nrb
This required parameter is the address of the NRB data area that is to be passed to NetEx.

buffer
This parameter required specifies the start of an array that holds the user data that is to be sent to the corresponding application.

length
This required parameter specifies the length of the data (in addressable units - words for Unisys) that is to be sent to the corresponding application, to be presented with the completion of the corresponding application's OFFER request. If no data needs to be sent to the other application, the length may be set to zero. The data type of length should be INTEGER.

datamd
This required parameter specifies the datamode to be used to send the connect data to the corresponding application. The data type of datamd should be INTEGER. Refer to NRBDMODE in "NetEx Request Block" for a discussion of the datamode parameter.

SCLOS Entry Parameters

The following NRB fields are used by SCLOS on entry.

NRBBUFA Address of outgoing Pdata.

NRBLEN Length of outgoing Pdata,

NRBUBIT Pdata unused bit count.

NRBDMODE Datamode of Pdata.

NRBPROTA Address of outgoing Odata.

NRBPROTL Length of outgoing Odata.

SCLOS Results

The following NRB fields are updated when SCLOS completes.

NRBSTAT Success/failure code.

SWAIT FORTRAN Call

The SWAIT call provides the means to wait for the completion of NetEx requests. Control will be returned to the SWAIT caller as soon as it is found that any one of the NRBs specified no longer has the "in progress" flag set. Return from the subroutine will be immediate if any one of the NRBs has completed before the SWAIT call.

After control is returned, it is the responsibility of the calling FORTRAN program to determine which of the NRBs in the list has completed. This can be done by examining the NRBSTAT field of each of the NRBs. Callers should note that more than one NRB may have completed before control is returned to the caller.

Call	Operation	Parameters

`CALL`	`SWAIT`	`(nrbnum,`
		`nrb`
		`[,nrb...])`

The following parameters were shown in the SWAIT call format. The parameters must be specified in the order presented. If parameters are omitted, the commas must still be used.

CALL

This is a required standard high level call instruction.

SWAIT

This is the verb for this call.

nrbnum

This required parameter indicates the number of NRBs to wait for. Control is returned after the completion of any calls/NRBs specified. If nrbnum is equal to 0, outstanding NRBs to the user are updated followed by an unconditional return. If nrbnum is equal to -1, wait for any request issued by this program to finish.

nrb

This parameter (required if NRBNUM >0) indicates the address of one or more NRBs (the number of NRBs specified in nrbnum) associated with the wait request. The nrb is not needed for SWAIT (-1) or SWAIT (0).

SDISC FORTRAN Call

The SDISC (disconnect) call provides the means for any connected application to terminate a session. The request is immediate and any data currently in transport may not be delivered. If data delivery is required, it is the responsibility of the disconnecting session to wait for confirmation of previous SREAD or SWRIT calls before issuing the SDISC. Before an SDISC can be issued, the user must provide an NRB with an NRBNREF relating to this session.

Call	Operation	Parameters

`CALL`	`SDISC`	`(nrb,`
		`buffer,`
		`length,`
		`datamd)`

The following parameters were shown in the SDISC call format. The parameters must be specified in the order presented. If parameters are omitted, the commas must still be used.

CALL
This is a standard high level call instruction.

SDISC
This is the verb for this call. Either SDISC or SDISCW must be specified.

SDISCW
This specifies that the calling program must wait for the call to complete before processing is resumed.

nrb
This required parameter is the address of the NRB data area that to be passed to NetEx.

buffer
This optional parameter specifies the start of an array that holds the user data that is to be sent to the corresponding application. Note that in the single case of SDISC, delivery of data is not reliable, although the actual disconnection will always occur.

length
This optional parameter specifies the length of the data (in addressable units - words for Unisys) that is to be sent to the corresponding application, to be presented with the completion of the corresponding application's SREAD request. If no data needs to be sent to the other application, the length may be set to zero. The data type of length should be INTEGER.

datamd (optional)
This parameter specifies the datamode that is to be used to send the disconnect data to the corresponding application. The data type of datamd should be INTEGER. Refer to NRBDMODE in "NetEx Request Block" for a discussion of the datamode parameter.

Upon completion of the SDISC, the connection will no longer exist. New commands against that connection will be rejected. An SOFFR or SCONN must be issued to establish a new connection.

SDISC Entry Parameters

The following NRB fields are used by SDISC on entry.

NRBBUFA Address of outgoing Pdata.

NRBLEN Length of outgoing Pdata.

NRBUBIT Pdata unused bit count.

NRBDMODE Datamode of Pdata.

NRBPROTA Address of outgoing Odata.

NRBPROTL Length of outgoing Odata.

SDISC Results

The following NRB fields are updated when SDISC completes:

NRBSTAT Success/failure code.

FORTRAN Requestor Program Example

Following is an example of a remote file access program. This program prompts the user for a record number, calls NetEx to obtain the information in the remote file, and displays a portion of the contents of the record on the user's terminal.

```
C*****************************************************************
C
C               Remote File Access Program
C
C   This program is designed to display records in the remote
C   file controlled by a direct access file service program.
C   The program prompts the user for the record number needed
C   in the remote file, calls NETEX to obtain the information
C   in the remote file, and displays a portion of the contents
C   on the user's terminal.
C
C*****************************************************************
C
C   FNRB is the 40 words needed for a NETEX request block.
C
        INTEGER FNRB(40)
C
C   NXDATA is the buffer of data exchanged with the remote
C   program. NXDATA(1) contains the request type;
C   NXDATA(2) contains the record number, and the remainder
C   contains the 1K bytes of file information on return.
C
        INTEGER NXDATA(258)
C
        DATA (FNRB(I),I=1,40) /40*0/
C
C   Connect to the program.    A password must be provided in
C   the first two words.   The remote program is assumed to
C   reside on host   `XYZ'.  If connect fails for any reason,  exit.
C
        NXDATA(1) = 1234567
        NXDATA(2) = 7654321
        CALL SCONNW (FNRB,NXDATA,2,0,'SERVER  ','LOOPBAK ')
        IF (FNRB(1).NE.0) GOTO 900
C
C   Read connect confirmation. If other than a confirmation
C   is received, disconnect and exit. If confirm is not received
C   within 30 seconds, exit.
C
        CALL SREADW (FNRB,NXDATA,256,30)
        IF (FNRB(1).NE.0) GOTO 800
        IF (FNRB(2).NE.2) GOTO 809
C
C   Process query
C
100     WRITE (6,110)
110     FORMAT (' Enter record number as S digits:')
        READ (6,120)  IREC
```

```
        IF (IREC .EQ.  0) GOTO 806
120     FORMAT   (I5)
C
C   Send request to remote program and wait for response.
C   Exit if response is not received within 30 seconds.
C
        NXDATA(1)=0
        NXDATA(2)=IREC
        CALL SWRITW (FNRB,NXDATA,2,0)
        CALL SREADW (FNRB,NXDATA,256,30)
C
C   Confirm that a data indication was returned and that the
C   NETEX read was OK; otherwise exit.
C
        IF (FNRB(1).NE.0) GOTO 800
        IF (FNRB(2).NE.3) GOTO 800
        IF (NXDATA(l).EQ.0) GOTO 240
        WRITE (6,220)
220     FORMAT (` NETEX read error.')
        GOTO  100
C
C   Display a portion of the returned    information
C   to verify read.
C
240     WRITE  (5,360) (NXDATA(I+2) ,I=1,24)
300     FORMAT (3(8(1X,I10)/))
        GOTO  100
C
C   End of run or abnormal result from NETEX.
C   Terminate the connection.
C
800     CALL SDISCW (FNRB,NXDATA,e,o)
900     STOP
        END
```

Installation

Overview

This part describes the installation of NetEx. The person installing NetEx must be an experienced systems programmer with access to configuration information for all machines that this host is to communicate with via the network.

This section describes the prerequisites to install NetEx, and a way to estimate the size of NetEx.

Prerequisites

NetEx is a software package that runs on Unisys OS2200 machines. The prerequisites to run NetEx are:

- A Unisys ClearPath/Dorado system running OS2200.
- CPCOMM configured as specified below.
- No Unisys software other than the base operating system and standard system processors is required to run NetEx. DAP is required to process NetEx dumps.
- You must obtain a valid license key for this product for each host/partition/siteid it will be run on and install it in the appropriate KEY file element (see Step 7. Install Software Key).

Using CPComm

Network interfaces may or may not be eligible for use by a CPComm process (an application program that signs onto CPComm using a CPCOMM_UID and CPCOMM_PWD). Our CPComm process is the IPC component of H300IPC. In your CPComm configuration file, several PROCESS statements are listed. When you install H300IPC, you will add a PROCESS statement for NETEXA. If IP addresses are not coded on a PROCESS statement, all network interfaces are available for use by that process. If IP addresses are coded on the PROCESS statement, that process may ONLY use those interfaces associated with the specified IP addresses. Multiple IP addresses may be assigned to the same network interface.

Once a pool of interfaces has been determined for a process (application program), how they are used is up to the specific application. Many IP programs (especially TCP programs) use a "LOCAL_IP_ADDRESS" of 0 when they make their calls to CPComm. This is a signal to CPComm that any interface available to that application maybe used to deliver the message. The application has no knowledge of which interface will actually be used.

In the Netex implementation, we assign a GNA address to an IP address. This can be done by using the SET IP command, or by using a DNS server. See Using DNS for details. Multiple GNAs may be assigned to one IP address and multiple IP addresses can be assigned to a network interface. Multiple IP addresses may NOT be assigned the same GNA. We will only use the interface for which GNA addresses are defined. When a Netex session is started, a Netex path is selected. This path contains the local GNA address. This GNA points to a specific IP address. When Netex makes a call to CPComm, the "LOCAL_IP_ADDRESS" is set to the IP address associated with that GNA. CPComm MUST honor that address. The network interface associated with that IP address is the ONLY interface CPComm can use. If the message fails to get delivered and acknowledged, Netex will go into APR and will select a new path. The request will be re-driven on the IP-address associated with the new local GNA address. If all sessions are started on the same path, only the interface associated with that IP address will be used.

When “ALTFIRST” is enabled, Netex will rotate through the paths when a session is started. (First session will start on path 1, second session on path 2, ….) This will have the effect of spreading your load across multiple interfaces available to that specific Netex application.

Using DNS

H300IPC may be configured to use a DNS server in your network. (This includes the file defined in the CPComm TCP/IP-DNR host-file statement.) To use the DNS server, NODNS=0 must be added to your NTX$INIT file. This instructs NetEx to use DNS to resolve the GNA to IP address mappings. This can be used in place of the SET IP commands. It also allows the installation to use one common set of GNA to IP address mappings across all copies of NetEx in the network.

When NetEx starts and NODNS=0 is in the NTX$INIT file, Netex will make a DNS request for each GNA address defined in the NCT. The request will be in the format “NTX0000nnnn” where nnnn is the GNA address. If this host is defined, your DNS server will return the assigned IP Address. It is recommended that when the NTX0000nnnn hosts are defined to your server, the NetEx host name is also included. This will assist you in future modifications to your network. See your network support specialist for specific instructions for adding DNS names to your server.

If you add the names to your CPComm host-file, it would look like this (using Unix format):

```
10.1.6.168  NTX00000A61.NETEXSW.COM  ALT1.NETEXSW.COM
```

This defines the GNA address 0A61 to an IP address of 10.1.6.168. It also defines ALT1.NETEXSW.COM to the same IP address. NETEXSW.COM was the default domain name coded in our CPCOMM configuration and is shown for clarification purposes only. Your default domain name would be different.

Installation Procedure

The installation steps are summarized here, then described more fully below:

1. Configure CPCOMM
2. Download the release file and check for required updates.
3. Insert user exits, modify NTX/map & rebuild NetEx (optional).
4. Do LOCAL INSTALL of Netex common banks
5. Create the network configuration table (NCT) and the configuration output file (PAMFILE).
6. Create an initialization file of non-default parameter settings.
7. Install Software Key
8. Start IPC & NetEx.
9. Verify the install.
10. Remap Applications

Each step is discussed in greater detail in this section. In the procedure that follows, it is assumed that NetEx will reside in the file NETEX, the configuration manager will reside in the file CONFIG, and the PASCAL compiler will reside in the file PASCAL. These need not be the actual file names.

If this is not the first install, it is recommended that you back up the current files before beginning the new install.

Step 1. Configure CPCOMM

A PROCESS statement must be added to the configuration for CPCOMM to provide a process-id and password for H300IPC. The IPC component will use this userid and password to communicate with CPComm. Up to six copies of NetEx may communicate with a single copy of the IPC component. The process-id defaults to **NETEXA**, and the default password is **NTXAPASS**. These maybe altered if you use the CPCOMM_UID and CPCOMM_PASS NetEx initialization parameter statements. If you wish to limit the NetEx network to a subset of the installed IP address, the IP Addresses statement would also need to be included. If the statement is not included, NetEx may use all the installed IP addresses. A sample is located in the release file element CPCOMM-CFG/SAMPLE. We also recommend setting INPUT-QUEUE-THRESHOLDS in the PROCESS statement as in the following example:

```
PROCESS,NETEXA  PASSWORD,NTXAPASS ;
   INPUT-QUEUE-THRESHOLDS,200,1000,1000000
```

If more than six copies of NetEx are required per operating system, please contact support. For CPComm failover and redundancy considerations, further advice on setting INPUT-QUEUE-THRESHOLDs or running multiple copies of CPComm, refer to the current Unisys "*Communications Platform Configuration and Operations Guide*" for ClearPath OS2200...

Step 2. Download Release File

The product is distributed as a file download. To download the distribution file, contact support at support@netex.com for download instructions. Regardless of the method used to receive the release file, it must be transferred to the target Unisys system via cpFTP using the "bin" and "quote site cfmt" transfer options.

Step 2a. Check for required updates.

Check if there are any updates by going to www.netex.com, and under the Support tab select products – follow the link to Unisys Clearpath/Dorado (H30x) platform and then select 'Updates' for the appropriate H30x product. If there are any, download them and follow their installation instructions.

Step 3. Insert User Exits & Re-MAP (optional)

Refer to the "User Exits" section for details. If user exits are not required, this step may be skipped.

To re-map the NetEx program banks, attach the name BUILD to a working copy of the NETEX release file and do:

- @ADD BUILD.CUSTOMER/REBUILD

to produce the new NETEX absolutes in BUILD.

Step 4. LOCAL INSTALL the Netex Common Banks

If you are upgrading to a new release, modify and run the element reinstall/sample. You will need to change the copy statements to the new release library, and the SYSLIB library to point to the library used when you did the SOLAR install. You should then reload each common bank from the updated library.

RL NTX$COMN

RL NTXMBXD

(**Note:** FOR initial installation)

The AFCBs NTXMBXB, NTX$COMN, & NTX$DBn must be installed by the Unisys Solar program. Sample SGSs for the local install are in "PROD-INSTALL/SAMPLE" in the NETEX release file.

Step 5. Create NCT and PAM Files

(**Note:** this step is required only at initial installation & when the network configuration is changed.)

The site systems programmer must prepare the network configuration statements described in "Configuration Management", or copy the NCT file from another system. These statements are used to describe the network configuration to NetEx.

The file NETEX$CONFIG must be available to NetEx when it executes. If it doesn't already exist, do one of the following:

```
@CAT,P  NETEX$CONFIG,F///500   -or-    @USE
NETEX$CONFIG,yourCONFIGfile
```

This file holds the network configuration output data (PAMs). This file is referenced directly by NetEx. Execute the configuration manager as described in "Using the Configuration Manager", to process the NCT and produce a PAM file.

Step 5a. Install Conversion Tables (optional)

If the site requires ASCII to EBCIDIC or EBCIDIC to ASCII conversion by H300IPC copy the following symbolic elements from the Netex Release file into NETX$CONFIG. The elements to copy are:

```
ASCIIEBCIDIC/TXT
EBCIDICASCII/TXT
```

The site may edit these text files to alter specific conversions that are performed. See the actual elements for the data format to use.

If these elements are not copied to NETEX$CONFIG a warning will be issued at IPC startup. IPC will function normally, except character conversion must not be requested.

Step 6. Create an Initialization File

If this is an initial install, the element "NETEXRUN/SAMPLE" in the NETEX release file should be modified to reflect site requirements (runid, account, etc.). The NTX and IPC runs share some parameters so a common copy should be maintained in the NETEX$CONFIG file element NTX$INIT. See "Initialization Statements". When editing this file, always save it as an ascii file, not as fieldata (@ed,uq file.NTX$INIT).

Step 7. Install Software Key

(**Note:** this step is required only at initial installation.)

In order for Netex to generate your key(s), you must provide the output from the execution of the SYSINFO utility supplied in the Netex Release file, e.g.:

```
@NetexReleaseFile.SYSINFO
Sysinfo: nnnnnnnnXXXXXXXXXXXX
```

It is suggested that you email your request, including the SYSINFO output, to (support@netex.com) and we will return your key(s) via email. Cut-and-paste the key data received into a "NETEX$CONFIG.$KEY$" file element. Edit the values into the "KEY" element in ASCII format with no leading spaces, e.g.:

```
@ED,IQ  NETEX$CONFIG.$KEY$
** NETEX/IP key for system ABCD **
B4MY-2AF4-AUFA-AALB-PBMR-6R2N-VHUO-VK4M
@eof
```

At initialization NETEX must have access to a file element with the name "NETEX$CONFIG.$KEY$" containing a valid product key record for NETEX on the executing host. Multiple keys for various products and hosts may be in the element. Any record beginning with "*" is treated as a comment and ignored.

Step 8. Start IPC & Netex

Refer to the samples, IPCRUN/SAMPLE and NETEXRUN/SAMPLE, for run requirements.

The IPCKEY value followed by "term" will terminate the IPC task and all NetExes associated with IPC.

Step 9. Verify Install

After installing and starting, test the functionality of NetEx by issuing a remote operator command to the local host name (set in the "NETEX$CONFIG.NX$INIT" file, and displayable with a DISPLAY PARMS operator command), which will cause an intrahost session to be established, (NX is your NetEx keyin, SYS2200A is your NetEx hostname) e.g.:

```
NX >SYS2200A DISPLAY ADAPTERS
```

Next, do the same command using host name NTXLCLnn which will use a hardware loopback path to the local host:

```
NX >NTXLCL01 DISPLAY HOST NTXLCL
```

Then do the same using a configured remote hostname (see NX D H output), e.g.:

```
NX >IBMPAR12 D S
```

Bring down NETEX and IPC, by entering the

```
IPC TERM
```

command on console. (IPC is the value of IPCKEY in the initialization file). Verify the DIAG$ file used in your IPC run stream is written to. You may use the @prt,f *IPC$DIAGfile.* This file will be required by NETEX support to diagnose problems that may occur.

Step 10. Remap Applications

Any applications that call Netex may need to be re-mapped to refer to the new bank BDIs and to remove S-options on BANK directives if you were previously using NCCBs.

NetEx Dumps

The NETEXRUN/SAMPLE shows how dumps are produced and are analyzed after Netex execution. F-cycles are used so NetEx can be restarted without destroying the previous dump file.

Error processing requires that the dump analysis functions be available in a file called NETEX, and it uses the DAP processor. The dump analysis omnibus functions must be available to be restored from TPF$.

MBX-peek

This program, which is executed as part of the dump analysis procedures after Netex execution, provides a formatted dump of the mailbox bank used to pass requests and status between Netex and IPC. It can be run at any time.

NetEx Bank Structure

NetEx is a real-time program with a non-configured common bank; thus it must reside and be executed from a registered (in the OS sysgen) common bank file. Most configurable NetEx parameters are read dynamically at program initialization.

The NetEx common I-bank (NTX$COMN) start address is 01000, which is also the address of the guaranteed entry point "NETEX". User programs should be careful to avoid overlapping this address with any bank that must be visible while NTX$COMN is based. User I-banks should start at 02000 or higher.

Other banks used by NetEx are as follows:

NTX$I

This contains all instructions and data except user interface (write-protected and not in the user's address space). This bank is based at 020000.

NTX$DBn

This is the common data buffer area for copying user buffers (normally based at 0300000). Different users may use different common data banks concurrently. Buffer sizes (initialization tag BUFSIZE) above 131000 require this bank starting address to be lowered.

NCTBANKn

These are dynamic banks used to hold configuration path data (based at 0740000). This bank is not in the user's address space.

NTXMBXD

This is a common data bank for communication between NetEx(es) and the IPC program. Currently it starts at 0771000.

User Bank (window)	Relative Start Address	NETEX Bank (window)
NTX$COMN (util-I)	01000	NTX$COMN (util-I)
NTX$I (main-I)	020000	NTX$I (main-I) (c. 16K)
NTX$DBn (util-D)	0300000	NTX$DBn (main-D) (variable-BUFSIZE)
NTXMBXD	0710000	MAILBOX bank (util-D)

Figure 11. NetEx Bank Window Structure

A user calling NetEx has his or her own utility I-bank replaced by the NetEx common bank (NTX$COMN). The original bank is replaced on return to the caller,

User Exits

NetEx contains the capability to have the installation Systems Programmer supply User Exits that can perform various statistical and validation functions specific to the user's environment. If a change needs to be made to one of these exits, the NetEx program must be rebuilt with the new, user-supplied, Exit routine. As released, the User Exit routines exist in element NXEXIT but only perform return jumps to their callers.

Currently, the User Call exit and the User Done exit are defined.

User Call Exit

This exit is located in the Request Manager Call processing code. At entry the following input is given:

```
A1 = NETEX Request Block address (NRB)
```

The exit may use A0-A5.

User Done Exit

This exit is located in the Request Manager Done processing code. It is entered when the request has been given a return status and is about to be given back to the User Interface. At entry the following input is given:

```
A1 = NETEX Request Block address (NRB)
```

The exit may use A0-A5.

Multiple NetExes

To build an alternate NetEx for testing, security, application isolation, etc., do the following:

- Create a working copy of the Netex release file, attach the USE name BUILD
- Edit the map elements to rename the common banks NTX$COMN & NTX$DB0 for the alternate Netex
- Add BUILD.BLD
- Install the new common banks

Applications using the alternate Netex must reference the new banks. The initialization file for the alternate Netex must specify the correct NTXNUM, NTXBDIS, HOST, and CONSKY.

Initialization Statements

The NetEx initialization statements supply information to NetEx regarding the local host and desired default values. Initialization statements form a data file that is assigned to the NetEx run at initialization time as described in “Installation Procedure”.

Following is a summary of the initialization statements. Each is described in detail in the succeeding paragraphs.

Keyword	Description
BUFSIZ	Specifies the size of the storage buffer size
CONSKY	Set the ER KEYIN$ prefix
CONTO	Connect timeout
CPCOMM_MODE	Specifies the CPCOMM Mode to use.
CPCOMM_PORT	Specifies the port number used by the Netex Network (6950)
CPCOMM_PWD	Password used to communicate with CPComm
CPCOMM_UID	Userid used to communicate with CPComm
DBANKS	Number of NTX$DBn banks used for user data buffers (default=1)
DEADTO	Disconnect (“deadman”) timeout
DEFBI	Default maximum input block size
DEFBO	Default maximum output block size
DREADQ	Number of driver reads to issue
DRIVMAX	Maximum number of concurrent DRIVER level user connections
HOST	Host Identifier
IPCKEY	ID string the operator uses to communicate with IPC
IPCUSE	@USE name for the IPC breakpointed print file
IDLTO	Idle timeout
LONGMSG	Enables long message support
LOGF	The file name of the netex breakpointed print file
LOGQ	The highlevel qualifier of the netex breakpointed print file
MAXBI	Maximum input block size
MAXBO	Maximum output block size
MAXDDBQ	Maximum blocks queued per DREF
MAXOD	Maximum user octet data allowed
MAXSEG	Specifies the maximum segment that will be sent over the network
MSGLVL	Default message level
NETMAX	Maximum concurrent network level user connections

NODNS	Specifies whether Netex will try to resolve GNA addresses to IP addresses via DNS lookup (default=0)
NTXBDIS	BDI of the NTX$COMN bank (NTX$DBn bank(s) follow consecutively
NTXMAIL	BDI of the Netex-IPC communication mailbox bank
NTXNUM	Index of an instance of Netex (for multiple concurrent NetExs)
NTXOPER	Maximum remote operator receivers active
PIOERR	Print I/O errors
READTO	Read timeout
ROPCLASS	Privilege level for remote operator requests
RTLEVEL	Relative real-time priority
SESMAX	Maximum number of concurrent sessions
SNDGRNM	SNDGRNM specifies when group names are used, whether the group name (1) or the real host name(0)
TRACE	Specifies if NetEx internal event tracing is to take place during execution of the NetEx process
TRANMAX	Maximum concurrent transport level user connections
TRCNUM	Number of buffers to allocate for tracing
TRCOFF	Types of events to be traced
TRCSIZE	Size of each trace buffer
USE	@Use name associated with the netex breakpointed print file
WDOGINT	Internal event ("watchdog") timing interval
XMTIMO	Used to set a timer for unterminated user programs (usually transactions) that need to be cleaned up because their completion was not detected by the user interface.
XNITS	Extra memory allocation for NetEx Internal Tasks
XnnnS	Extra memory allocation for miscellaneous control blocks
XXMUBS	Extra memory allocation for cross-memory user blocks

General Form of Initialization Statements

All initialization statements take the following form:

Keyword	Operands
KEYWORD	= OPERAND . comments

KEYWORD

This starts in the first character position of the statement and ***must be at least six characters long (including trailing blanks).*** Only one keyword is allowed in each initialization statement.

OPERAND

This consists of keyword-dependent information which is separated from the keyword by an equals (=) sign.

comments

Comments may be inserted after the operands. A period must precede the comments.

The initialization statements may be supplied in any order. If a required statement or operand is missing, NetEx will produce a diagnostic message and abort. The only initialization statement that is currently required is the HOST statement. All others default to pre-determined values.

The following paragraphs contain descriptions of all of the initialization statements.

BUFSIZE Initialization Statement

The BUFSIZE statement specifies the size of data buffer storage to be used in this NetEx. This amount of storage will be allocated as a contiguous block (one per D-bank) in memory and will be sub-allocated for the different buffering requirements of the NetEx applications. Choosing a buffer size too small may reduce NetEx performance; choosing a buffer size too large has little effect on NetEx performance, however, system performance may degrade due to shortage of memory. Buffer space usage can be monitored with the operator request DISPLAY MEMORY.

The BUFSIZE statement sets the size of each of the NetEx D-banks. Normally there is only one D-bank configured (see configuration statement DBCNT). If the requested size is greater than the space between the start of the D-bank and the end of memory, the size will be reduced to what is available. If more space is needed, the starting address of the D-bank will have to be reduced.

Keyword	Operands
BUFSIZE	= words

words

Indicates the number of words of buffer space to reserve for D-bank 0,1,...,n. (Unless the start address of the NTXDB$n banks is adjusted downward, the upper limit of WORDS is 131,000.)

If omitted, the largest size that will fit in the address space is used.

CONSKY Initialization Statement

The CONSKY statement sets the ER KEYINS, so it can be used for operator or user communication, rather than COM$ (or TREAD$ in demand mode). CONSKY may be the desired keyin id or 1 (id = 'NTX') or 0 (use COM$/TREAD$). The NetEx userid must have SSCONSOLE privilege, and requestors must have at least FULL CONSMODE for class A keyins, LIMITED for class C keyins, and BASIC for class G keyins

Keyword	Operands
CONSKY	= 0 | 1 | keyin

If omitted, CONSKY = 0 is assumed.

CONTO Initialization Statement

The CONTO statement specifies the maximum number of seconds that NetEx will wait for a Transport Connect message to generate any response from the remote host. If this time is exceeded, the transport protocol will assume that the remote host or remote NetEx is "down" and return a Disconnect Indication to the transport caller. If the caller has specified Session connection service, then Session will retry another connection on an alternate path if one is available. (It might be useful to set CONTO to a lower value if alternate pathing is available, in order to invoke it sooner). Note that CONTO should be a greater value than IDLTO (described below) so that the responding transport protocol can respond with an "idle" message if the delay is due to the responding application program.

Keyword	Operands
CONTO	= seconds

seconds

Indicates the number of seconds that NetEx will wait for a CONNECT response.

If omitted, CONTO = 30 is assumed.

CPCOMM_MODE Initialization Statement

The CPCOMM_MODE statement allows a user to specify which CPCOMM Installation Mode should be used with H300IPC. The value of this parameter must be one of: A, B, C, D, E, F, G or H. The default installation mode used will be "A" if this parameter is not provided.

The installation mode to be used will be reported in an IPC informational message. For example:

```
IPC I 0204@121812.392 Requesting CPComm Mode: D
```

If a Mode is specified that is not actually installed, IPC will abort with a Common Bank error. For example:

```
IPC I 0204@121812.392 Requesting CPComm Mode: E
Common bank error 9 . -1->
-1-> common bank " UNDETERMINED " with bdi 405225
-1-> referenced by run-id " SVNBLD " is not installed.
```

Keyword	Operands

CPCOMM_MODE	= CPCOMM installation mode

If omitted, CPCOMM_MODE=A is assumed.

CPCOMM_PORT Initialization Statement

The CPCOMM_PORT statement allows a user to specify which IP PORT should be used with H300IPC. The value of this parameter must be consistent throughout your Netex network. The default value is 6950.

Keyword	Operands
CPCOMM_PORT	= number

If omitted, CPCOMM_PORT = 6950 is assumed.

CPCOMM_PWD Initialization Statement

The CPCOMM_PWD statement allows a user to specify the password required to access CPCOMM. The value of this parameter must match the value coded in the CPComm configuration file. The default password NTXAPASS will be used if this value is not coded. Valid passwords are one to twenty alphabetic or numeric characters.

Keyword	Operands
CPCOMM_PWD	= password

If omitted, CPCOMM_PWD = NTXAPASS is assumed.

CPCOMM_UID Initialization Statement

The CPCOMM_UID statement allows a user to specify the userid required to access CPCOMM. The value of this parameter must match the value coded in the CPComm configuration file. The default userid NETEXA will be used if this value is not coded. Valid userids are one to ten alphabetic or numeric characters.

Keyword	Operands
CPCOMM_UID	= userid

If omitted, CPCOMM_UID = NETEXA is assumed.

DBANKS Initialization Statement

The DBANKS statement specifies the number of NTX$DB0 banks to use for user data buffering. Applications must be specifically MAPed to use these banks.

Keyword	Operands
DBANKS	= 1 .. 10

If omitted, DBANKS=1 is assumed.

DEADTO Initialization Statement

The DEADTO statement specifies the amount of time that Transport Software will wait until it assumes a disconnection because the remote host does not communicate. The local host transport software will generate an "idle" message every IDLTO seconds; the remote NetEx transport software, if operational, will generate an "idle" message of its own. If any message whatsoever is received from the remote host, DEADTO is reset. When DEADTO expires, the transport connection terminates.

Keyword	Operands
DEADTO	= seconds

seconds
Specifies the number of seconds that NetEx will wait before disconnecting a dead session.

If omitted, DEADTO = 60 is assumed.

DEFBI Initialization Statement

The DEFBI statement specifies the default maximum input block size for a Transport connection. If the user specifies an incoming block size parameter in an SOFFER or SCONNECT request, then the specified size is granted. Otherwise, the default of DEFBI is used.

Keywords	Operands
DEFBI	= words

words
Represents the default incoming block limit in words.

If omitted, DEFBI = 512 is assumed.

DEFBO Initialization Statement

The DEFBO statement specifies the default maximum output block size for a Transport connection. If the user specifies an outgoing block size parameter in a SOFFER or SCONNECT request, then the specified size is granted. Otherwise, the default of DEFBO is used.

Keyword	Operands
DEFBO	= words

words
This is the default outgoing block limit in words.

If omitted, DEFBO = 512 is assumed.

DRIVMAX Initialization Statement

The DRIVMAX statement specifies the maximum number of concurrent Driver Interface connections supported by NetEx. This count reflects only the number of DCONNECT's in effect. If a new DCONNECT is issued when DRIVMAX sessions are in progress, the DCONNECT request will be rejected with an error.

Keyword	Operands
DRIVMAX	= number

number
This is the number of concurrent connections supported.

The default is DRIVMAX = 0.

DREADQ Initialization Statement

The DREADQ statement specifies the number of concurrent reads issued by the Netex driver on each adapter input port.

Keyword	Operands
DREADQ	= number

number
This is the number of driver reads to issue (range 3-100).

The default value is 5.

HOST Initialization Statement

The HOST statement identifies the host processor that this current version of NetEx is working with. The hostname specified must match exactly the hostname used in other installation statements (see Configuration Manager HOST label).

Keyword	Operands
HOST	= host-name

host-name
This is a one to eight character name.

This statement is required. There is no default.

IDLTO Initialization Statement

The IDLTO statement specifies the amount of time that transport protocol will wait before sending an "idle" message to verify the continued existence of the communicating party at the other end of the connection. The time between idle messages is specified in seconds.

Keywords	Operands
IDLTO	= seconds

seconds
This is the number of seconds that NetEx will wait before sending an idle message.

If omitted, IDLTO = 5 is assumed.

IPCUSE Initialization Statement

The IPCUSE statement specifies the USE name in the IPC run procedure associated with the breakpointed print file.

Keywords	Operands
IPCUSE	= name

name
This is the use name associated with breakpointed print file for use with IPC. You must specify this name before the IPC SWITCH command is enabled.

IPCKEY Initialization Statement

The IPCKEY statement defines the console key in string to communicate with IPC.

Keywords	Operands
IPCKEY	= key-in

key-in

This is the console key-in string to communicate with IPC. The only valid commands are TERM, SWITCH and ABORT. ABORT and TERM cause all copies of Netex attached to this IPC to terminate.

IPC TERM – Normal termination
IPC ABORT – Terminate in error
IPC SWITCH – Create and use a new cycle for the print file.

If omitted, IPCKEY = IPC is assumed.

LOGF Initialization Statement

The LOGF statement specifies the filename to use when creating a new cycle of the netex breakpointed print file. You must also configure the USE and LOGQ parameters.

Keywords	Operands
LOGF	= filename

filename

The filename for the breakpointed print dataset.

LOGQ Initialization Statement

The LOGQ statement specifies the High Level Qualifier to use when creating a new cycle of the netex breakpointed print file. You must also configure the USE and LOGF parameters.

Keywords	Operands
LOGQ	= HLQ

HLQ

The high level qualifier for the breakpointed print dataset.

LONGMSG Initialization Statement

The LONGMSG statement enables long message support.

Keywords	Operands
LONGMSG	= 1

LONGMSG

This statement enables long message support. It may be set to any numeric value in the range of 0 to 999. Long message reduces the number of messages that must traverse the network when the user's application uses O-Data. The NCT must also have an OPTION=LONMSG statement coded for the local host name and for the remote host name that also supports long messages. On a running system, this can be verified by a "DISPLAY HOST *nnnnnnn*" command. The flags field is a sum of all flags set. This option has the value 4.

MAXBI Initialization Statement

The MAXBI statement specifies a maximum on the amount of data that may be received in a single message by this NetEx machine. An attempt to establish a session with a larger blocksize will be rejected. The principal use of MAXBI is to reduce fragmentation or excessive use of NetEx buffer memory.

Keyword	Operands
MAXBI	= words (0-65535)

words

This is the maximum incoming block size in words.

If omitted, MAXBI = 2048 is assumed.

MAXBO Initialization Statement

The MAXBO statement specifies a maximum on the amount of data that may be sent in a single transmission by this NetEx. The principal use of MAXBO is to reduce fragmentation or excessive use of NetEx buffer memory.

Keyword	Operands
MAXBO	= words (0-65535)

words

This is the maximum outgoing block size in words.

If omitted, MAXBO = 2048 is assumed.

MAXDDBQ Initialization Statement

The MAXDDBQ statement specifies the maximum number that may be queued per DREF. Driver data buffering allows data received for driver level users to be queued if there is no DREAD queued when the data is received. The user specifies the (unchangeable) datamode for input data when doing DCONNECT.

Keyword	Operands
MAXDDBQ	= number

number

This is the maximum number of input blocks queued per DREF. The range is 0 to 65536.

If omitted, MAXDDBQ = 5 is assumed.

MAXOD Initialization Statement

The MAXOD statement specifies the maximum number of octets of information a SESSION user may specify in the NRBPROTL field in the NRB. This is used to send user protocol information.

The maximum value is larger for TRANSPORT, NETWORK, and DRIVER level callers. It is increased by the maximum size of the previous level's protocol. This value is normally set to about 256, but can be set to whatever value the application user needs (consistent with buffer usage).

Keyword	Operands
MAXOD	= bytes (0-256)

bytes

This is the maximum amount of user protocol data allowed.

If omitted, MAXOD = 256 (the maximum) is assumed.

MAXSEG Initialization Statement

The MAXSEG statement specifies the maximum segment that will be sent over the network. This value is negotiated with the connected host so that the minimum segment for the two hosts is used.

If a session block that is larger is sent with SWRITE, Netex divides it into segment size pieces to send.

Keyword	Operands
MAXSEG	= words (0–65536)

words

The maximum segment size in words

If omitted, MAXSEG = 20000 is assumed.

MSGLVL Initialization Statement

The MSGLVL statement is used to set the default message level (MSGLVL) at the beginning of a NetEx run. This message level is used to determine which NetEx messages are displayed to the operator. The message level is a value from 0 to 15; 0 denotes the greatest level of detail (for diagnostic use only), and 15 indicates total errors.

Keyword	Operands
MSGLVL	= minimum-importance-level

minimum-importance-level

This specifies the minimum level of messages to be displayed (0 to 15 decimal). All messages that are more severe will also be displayed.

If omitted, MSGLVL = 8 is assumed.

NETMAX Initialization Statement

The NETMAX statement specifies the maximum number of concurrent network interface connections allowed by NetEx. This count reflects only the number of NCONNECTS or NOFFERS in effect. If a new request is issued when NETMAX number of sessions are in progress, the request will be rejected.

Keyword	Operands
NETMAX	= integer

integer

This is the maximum number of network connections allowed.
If not specified, NETMAX = 0 is assumed. The range is 0 to 65536.

NODNS Initialization Statement

The NODNS statement specifies whether Netex will try to resolve GNA addresses to IP addresses via DNS lookup or rely solely on SET IP commands. If zero, DNS lookup will be used at initialization and at LOAD NCT time. Note that any addresses returned by DNS lookup will NOT be used if there is a SET IP for the GNA.

Keyword	Operands
NODNS	= 0|1

A value of zero means ‘use DNS’; one means ‘no DNS’ (default)

NTXBDIS Initialization Statement

The NTXBDIS statement defines the BDI of the NTX$COMN bank if AFCBs are used for the Netex common banks. If this statement is omitted, NCCBs are assumed. NTX$DB0 bank(s) must follow this consecutively (see DBANKS Initialization Statement above.)

Keyword	Operands
NTXBDIS	= bdi

bdi

is the bank descriptor index of NTX$COMN (0-07777). The range is an octal 300 to octal 7777. There is no default value.

NTXMAIL Initialization Statement

The NTXMAIL statement is required. It must specify the BDI of the Netex-IPC communication mailbox bank on the local system

Keyword	Operands
NTXMAIL	= bdi

bdi

is the bank descriptor index of NTXMBXD (300-07777). The range is an octal 300 to octal 7777. There is no default value.

NTXNUM Initialization Statement

The NTXNUM statement is optional. It must specify the index number for a specific instance of multiple concurrent NetExes. 0 is assumed if omitted.

Keyword	Operands
NTXNUM	= num

num

is the Netex index value (0-5). There is no default value.

NTXOPER Initialization Statement

The NTXOPER statement specifies the maximum number of concurrent remote operator requests that NetEx will accept. Setting this greater than zero allows remote operators to request the status or modify the state of NetEx (see ROPCLASS below). It also allows local programs to connect and act as the NetEx operator to a limited extent.

The NTXOPER controlled services do not require facilities when not actually in use on this NetEx, and so little cost is incurred by setting this greater than zero. Normally, a setting from 2 to 5 is chosen.

Keyword	Operands
NTXOPER	= integer

integer

This is the maximum number of remote operator connections allowed.

If not specified, NTXOPER = 0 is assumed. The range is 0 to 65535.

PIOERR Initialization Statement

The PIOERR statement prints I/O errors to the NetEx print file when set to nonzero. When NetEx is taken down, the printout can be used by the maintenance people. The printout contains two lines with about the same information as the console display message from LIOERR.

The following are the two lines of print produced by the error shown under "M RESPONSE" and "UNSOLICITED CONSOLE MESSAGES".

```
DINIT: HYPER2 FUNCTION REJECTED  EXEC STATUS:00 SUBSTATUS:00
FUNCTION = 007000000000 EI 003000100000               12:49:45.387
DINIT: HYPER2 001000617731 400057125520 000000000000 003000100000
              000000000000 000000000000 000000000000 00000000000 12:49:45.387
```

All the fields are the same as in the console output except the CSW line does not have a header. For a non-word channel error, the printed error information is as follows:

```
DINIT: HYPB0  DEVICE NOT AVAILABLE  EXEC STATUS:07
SUBSTATUS:l0  FUNCTION=23  13:42:47.941
DINIT: HYPB0  001000610000 000057117720 000404000001 00000000000
              000000000000 000000000000 000000000000 00000000000
```

If this error contained a unit check and associated sense, the sense bytes would be displayed after the function.

Note that the error under CONSOLE IO ERROR MESSAGES is a communication type error and does not have anything printed via PIOERR or logged via LIOERR.

Keyword	Operands
PIOERR	= 0 | 1

0 (off) 1 (on)

If not specified, PIOERR = 0 is assumed. Once activated, this option can only be cancelled by re-initializing NetEx.

READTO Initialization Statement

The READTO statement specifies the amount of time that Transport software will retain data if the user does not issue a READ request to accept incoming data, Connect Indications, or Disconnect Indications. If Transport detects that data is to be sent to the user, it will wait for READTO seconds. If the data is not sent by that time, then Transport will cause a Disconnect to be sent to the corresponding party. All of the waiting data in the local host that was intended for the timed out application is discarded.

If a Disconnect Indication was not already waiting to be sent to the application, then Transport will hold the newly generated Disconnect Indication for another READTO seconds before dropping the Disconnect Indication. If the application issues a read after this time, it will receive the "T-Ref invalid" response indicating that Transport has no record of the transport connection.

Keyword	Operands
READTO	= seconds

seconds

This specifies the number of seconds that NetEx will wait before discarding waiting data.

If omitted, READTO = 30 is assumed. The range is 0 to 65,535.

ROPCLASS Initialization Statement

The ROPCLASS statement specifies the access level a remote operator may have in NetEx. There are three possible classes. For more information see the "Remote Operator Command Classes" section on page 114.

Keyword	Operands
ROPCLASS	= class

class

This indicates the maximum access allowed to a remote operator [A, C, G].

If not specified, ROPCLASS = G is assumed.

RTLEVEL Initialization Statement

The RTLEVEL statement allows the installation to control the relative priorities of NetEx and other real-time tasks in the host. The main NetEx activities will run at real-time level RTLEVEL and certain others (e.g., interrupt detection, operator command) will run at RTLEVEL-1 (i.e., one level higher on switching queues). NetEx will function if RTLEVEL = 0, but timing, etc. may be unpredictable.

Keyword	Operands
RTLEVEL	= integer

integer

This is the switching level for NetEx real-time activities

If omitted, RTLEVEL = 27 is assumed. The range is 0 to 65535.

SESMAX Initialization Statement

The SESMAX statement specifies the maximum number of concurrent Session connections supported by NetEx on this host. The count of connections includes both outstanding SOFFER's and completed Session Connections. If a new Session request is made when SESMAX Sessions are in progress, the SOFFER or SCONNECT request will be rejected with an error.

Keyword	Operands
SESMAX	= integer

integer
This is the maximum number of active sessions and offers that may be outstanding.

If omitted, SESMAX = 8 is assumed. The range is 0 to 65535.

SNDGRNM Initialization Statement

The SNDGRNM specifies when group names are used, whether the group name (1) or the real host name (0) should be sent on connects. The use of this parameter will depend on the use of group hosts in the NCT in the site configuration.

Note: BFX HOSTCHK will result in a failure if SNDGRAM is 1 or not set

eyword	Operands
SNDGRNM	= 0 | 1

integer
Interger maybe 0 for off or 1 for on.

If omitted, SNDGRNM = 1 is assumed.

TRANMAX Initialization Statement

The TRANMAX statement specifies the maximum number of concurrent transport interface connections allowed by NetEx. This count reflects only the number of TCONNECTS or TOFFERS in effect. If a new request is received when TRANMAX number of sessions are in progress, the request will be rejected.

Keyword	Operands
TRANMAX	= integer

integer
This is the maximum number of active connections that may be outstanding.

If not specified, TRANMAX = 0 is assumed. The range is 0 to 65535.

TRACE Initialization Statement

The TRACE statement specifies if NetEx internal event tracing is to take place during execution of the NetEx process. Three options are available:

- If TRACE = OFF, trace buffers are allocated but no tracing will take place.

- If TRACE = MEMORY, the buffers are allocated as described in the TRCNUM and TRCSIZE statements. However, the buffers are not written anywhere and events are traced circularly in the buffers.
- If TRACE = TAPE, fixed blocks of size TRCSIZE are written to file NETEX$TRACE which must be pre-assigned to the NetEx RUN.
- If TRACE = DISK, TRCSIZE must be a multiple of the sector size (28 words). Records are written to the file NETEX$TRACE. This file must be assigned to the netex run. TRCNUM should be set to 2.

The TRACE statement has the following format.

Keyword	Operands
TRACE	= | OFF | MEMORY | DISK | TAPE [UNL|UNLSTAR|LAB|CART] |

OFF trace buffers are allocated but no tracing is to take place initially.

MEMORY trace buffers are allocated; events are traced in memory circularly.

TAPE trace buffers allocated; events are traced and written to the tape file NETEX$TRACE. Tape options are: UNLabelled, UNLabelled STAR, LABelled, and CARTridge.

DISK trace buffers allocated; events are traced and written to the disk file NETEX$TRACE. When the file is full, it wraps back to the beginning. When tracing to disk, TRCNUM should be set to 2. TRCSIZE may be set to 8176. This must be a multiple of the 28 word sector size on disk. You also must include an @ASG,A NETEX$TRACE statement in your NetEx proc. When the trace file is printed, using the trc member of the program file, the blocksize must be changed to match the TRCSIZE size used.

If omitted, TRACE = MEMORY is assumed.

TRCNUM Initialization Statement

The TRCNUM statement specifies the number of buffers to be allocated for tracing purposes. The size of each trace buffer is specified by the TRCSIZE initialization statement. With a MEMORY trace, each of the buffers is filled in a circular fashion so that the oldest buffer is the one that is being filled. With a TAPE trace, I/O is scheduled to write a buffer as soon as the buffer is full. Sufficient buffers or buffer size should be allocated so that trace information is not lost by wrapping the trace buffer(s) while they are being written.

Keyword	Options
TRCNUM	= integer

integer

This specifies the number of separate trace buffers.

If omitted, TRCNUM = 8 is assumed. The range is 0 to 65535.

TRCOFF Initialization Statement

The TRCOFF statement is used to control which types of events are to be internally traced and which are not. By default, all event types are traced. The installation systems programmer may elect to suppress certain types of events to reduce the traffic on the trace file. Specifying a duplicate event will error the entire parameter line.

Each of the event types has a specific option:

MC Module Calls; includes called entry name and passed parameters

MR Module Return; traces flow of control back to a calling module

QM Queue Manager entries; records queue, queuing type and address of the entity moved

DO Driver Output; records all message propers sent to the network

DI Driver Input; records all message propers received from the network

DA Driver Associated Data (40 bytes) is recorded with DI or DO

IR I/O Request; records all I/O packets and channel programs issued against local adapter units

IC I/O Completion; records I/O packets at completion

UR User Request; records contents of every User Request Block coming into NetEx

UB User Buffer; records ;the memory buffer copy by the User Interface

UD User Done; records contents of every User Request Block returned back to the user

UC User Complete; occurs when the user has accepted the completed request

UT User Termination; records user termination automatic notification and cleanup by NetEx

BA Memory Buffer Allocations

BF Memory Buffer Frees

SP Spawn; records creations of new NITs

SH Trash Records; Messages that have been received and discarded

SU Suspend; records that a NetEx task has suspended voluntarily

OI Operator Input; records input to NetEx from operator

OO Operator Output; records all output from NetEx to operator

Keyword	Operands
TRCOFF	= event, event, ...

event

This is a 2-letter abbreviation of a trace option listed above

If omitted, all events are traced unless TRACE = OFF was specified.

TRCSIZE Initialization Statement

The TRCSIZE statement specifies the size of each trace buffer in words. Trace records are accumulated chronologically in the trace buffer. When a buffer is full, the next trace buffer begins to fill.

Keyword	Operands
TRCSIZE	= words

words

This is the size of each trace buffer in words.

If omitted, TRCSIZE= 1400 is assumed. The range is 99 to 34,000.

USE Initialization Statement

The USE statement specifies the USE name when creating a new cycle of the netex breakpointed print file. You must also configure the LOGQ and LOGF parameters.

Keyword	Operands
USE	= USE_NAME

USE_NAME

Specifies the name to use on the BRKPT of the netex print file.

WDOGINT Initialization Statement

The WDOGINT statement specifies the number of seconds elapsed between events timed by the Low Resolution (Watchdog) timer. This timer is used by many internal tasks in NetEx to allow the work in progress to abort if substantial delays occur. It is used by the CONTO, IDLTO, and READTO statements described in this section. Every WDOGINT seconds, the Low Resolution timer code is activated and decrements the elapsed time available for all internal tasks. Any time expiration, such as CONTO, must necessarily take place after an interval that is a multiple of WDOGINT.

Keyword	Operands
WDOGINT	= seconds

seconds

This is the number of seconds for the Low Resolution (Watchdog) timer.

If omitted, WDOGINT is set to 2. The range is 0 to 65535.

XMTIMO Initialization Statement

The XMTIMO statement is used to set a timer for unterminated user programs (usually transactions) that need to be cleaned up because their completion was not detected by the user interface. Termination is triggered if no user requests have been issued by the user for longer than this time.

Keyword	Operands
XMTIMO	= seconds

seconds

This is the time interval that the timer will use to terminate idle users.

If omitted, no timing occurs (default). The range is 0 to 65535.

XNITS Initialization Statement

The XNITS statement is used to allocate extra memory to hold NIT's, the central control blocks for NetEx Internal Tasks. Normally, NetEx initialization calculates a default number of NITs based on the SESMAX, MAXTRAN, and MAXDRIV parameters specified during initialization. Special considerations may require extra NITs be allocated. If NetEx has insufficient NITs, it will abort.

Keyword	Operands
`XNITS`	`= integer`

integer

This is the number of extra NITs required.

If omitted, no extra NITs are allocated. The range is 0 to 65535.

XnnnS Initialization Statement

XnnnS statements are used to allocate additional memory for storage used for miscellaneous control block storage by NetEx. Normally, NetEx initialization calculates a default number of buffers for each of these pools based on the SESMAX, TRANMAX, and DRIVMAX parameters. If extra buffers are required, they may be allocated using the appropriate XnnnS = parameter. The only reasonable way to know if these are needed is to observe the usage of these buffer pools during operation using DISPLAY MEMORY.

Keyword	Operands
`X32S | X64S | X128S`	`= integer`

integer

This is the number of extra control blocks required for this buffer size.

If omitted, no extra control blocks are allocated. The range is 0 to 65535.

XXMUBS Initialization Statement

The XXMUBS statement is used to allocate extra cross-memory user blocks for communication between application programs and the Netex user interface. Normally, NetEx initialization calculates a default number of XMUBs based on the SESMAX, MAXTRAN, and MAXDRIV parameters specified during initialization. Special considerations may require extra XMUBs be allocated. If NetEx has insufficient XMUBs, some applications may receive RC 503.

Keyword	Operands
XXMUBS	= integer

integer

This is the number of extra user blocks required. (1-200)

If omitted, no extra XMUBs are allocated.

Configuration Management

NetEx provides a configuration manager that is used to describe the topology of the entire network. One configuration file defines the network for all hosts, so that there is no need to generate different files for each host.

Using the Configuration Manager

The Configuration Manager program (CONFMANG) is executed by the user "off-line" to NetEx. The output of this process is a file of PAMs (Physical Address Maps, or network routes) that will be read by NetEx at initialization time, and also whenever the operator issues a LOAD NCT command.

The output file, which may be an SDF file or program file element, is referred to as the PAMfile. The input file, or element, describing the network is referred to as the NCT, or CONFFILE.

The following describes how to use the Configuration Manager.

1. Create or update the CONFFILE. Use the Configuration statements described below. It may be helpful to refer to the sample configuration in the file CONFIG.

2. Execute CONFMANG, as follows:

   ```
   @xqt[,s]   config.CONFMANG
   ```

   The program prompts for the next command (NCT, DESELECT, EXIT, etc.). The command must be in uppercase letters.

   Note: The keyword ECHO may be used to display input lines as they are read. NOECHO disables echoing. ECHO may also be set by using the S option.

3. Enter the following: NCT CONFFILE. or NCT file.CONFFILE

   This starts the configuration processor using the NCT you specified (CONFFILE). If you receive errors as a result of this step, enter EXIT to leave the configuration manager, correct the errors in your configuration file, then return to Step 2.

   The configuration manager automatically generates all possible loopback paths (paths out one local adapter and in another). The host name given to the PAMs with all the loopback paths is NTXLCL. A PAM is also created for each individual loopback path. The host name given to each one of these is NTXLCLnn, where nn is in the range 00 to 99. NTXLCL may be SELECTed or DESELECTed.

   Note: A warning message (CONF060E) is issued if there are no possible loopback paths. The message reads as follows:

   ```
   No path from host hhhhhh to NTXLCL
   ```

4. If all hosts and groups defined in the configuration statements are desired as destinations, omit this step. If only selected hosts are to be defined to NetEx (allowing them to be used as destinations), list them using one or more SELECT commands. Enter the following:

   ```
   SELECT hostname hostname ...
   ```

   This command names NetEx hosts (identified in the configuration statements) for which paths are to be generated. If the SELECT command is omitted, or if SELECT * is entered, all hosts are used.

   If a small number of host destinations are to be omitted, enter:

   ```
   DESELECT hostname.hostname ...
   ```

5. To generate the paths, enter:

   ```
   MAKEPAM hostname pamfile.hostname
   ```

   The hostname is the name of the “from host” or group (i.e., the host where the output file produced by this process will be used). All paths generated will be printed. The output of this step is placed in omnibus element hostname in the file PAMFILE, which must be made available to NetEx as NETEX$CONFIG.

   On the local system, hostname must match that on the HOST initialization statement. When NetEx is started, or LOAD NCT is requested, the omnibus element NETEX$CONFIG.host is accessed.

6. To exit the configuration manager program: EXIT

Configuration File

The configuration file contains the configuration manager statements that describe the user's network. Ten statement types are used to describe this:

VERSION - specifies the version of the network configuration. This value is recorded in the configuration record when the PAM file is built.

LOCALNET - describes all equipment that is interconnected via one or more networks. Statements describing the equipment on that network follow the LOCALNET statement.

> TRUNK – For IP networks, TRUNK specifies a name used to identify connectivity to an IP network.

HOST - describes a host processor that has a connection to the network via one or more processor adapters.

ADAPTER - specifies the address and characteristics of the processor adapter that is attached to the HOST. This typically refers to the Network Interface port on the HOST.

END specifies the end of the network configuration.

The syntax rules for these statements are as follows:

- Statements can only be up to 64 characters in length.
- The ASCII tab character is not recognized.
- All reserved words MUST be in upper case. A reserved word is the name of a statement (e.g., LOCALNET) or a parameter (e.g., TYPE).
- All references to identifiers MUST be identical to the identifier. The same combination of upper and lower case must be used (e.g., TO = Beta references the label “Beta” NOT “BETA” or “beta” or any other combination).
- If a label is present, it must begin in the first character position of the statement, with no leading blanks. At least one space must separate the label from the statement type and the statement type from the parameters. If a label is missing, at least one blank must precede the statement type.
- A comma “,” or a blank is used as separators within a line to delimit the parameters of each statement.
- Continuation statements are denoted by at least one blank preceding the statement.

- If an asterisk is detected in the first character position of the statement, the entire statement is treated as a comment.
- The beginning of an in-line comment is identified by an asterisk (*). The portion of the line from the * to the end is ignored. A line may be longer than 64 characters only if positions 64 to 80 are part of an in-line comment initiated by an *.

VERSION Statement

The VERSION statement specifies a site-dependent number assigned to the configuration file. Valid values are 0 through 255. If used, it must be the first statement in the file.

The VERSION statement has the following format.

Name	Statement	Parameters
	VERSION	nnn

The following control words are used in the VERSION statement.

VERSION
This is the verb for this statement,

nnn
This operand contains optional value assigned to the configuration file.

LOCALNET Statement

The LOCALNET statement defines the name of the local network. The term “local network” signifies Hosts, Adapters and Trunks sectioned into logical groups and separated by one of the high speed communications links. The first statement in the configuration file must be a LOCALNET statement. All TRUNK, HOST, and ADAPTER, statements for that local network must follow the LOCALNET statement. The presence of a second LOCALNET statement, regardless of the label, will begin the description of a second local network. At least one LOCALNET statement must be present in any network configuration.

The LOCALNET statement has the following format:

Name	Statement	Parameters
[label]	LOCALNET	TYPE=HC

The following control words are used in the LOCALNET statement.

label
This optional label specifies the name of this local network. This label should be used to make the NCT more readable. The label may be any name desired by the user which is one to eight alphanumeric characters long. It must be unique from all other labels in the network configuration. A typical label would be the site ID of the network which is referenced by remote sites.

LOCALNET
This is the verb for this statement.

TYPE
This required parameter specifies the type of local network to be described. HC stands for HYPERchannel network. . ***NetEx/IP networks must be defined as TYPE=HC.***

TRUNK Statement

The TRUNK statement specifies a name used to identify connectivity to an IP network. One trunk statement must be present for each network. Hardware connected to the trunk is identified in subsequent HOST statements. All TRUNK statements in a HYPERchannel local network must immediately follow the LOCALNET statement and precede all HOST statements that define the usage of the trunks. The range of a trunk is a single local network. A TRUNK defined in one LOCALNET may NOT be referenced in another.

The TRUNK statement has the following format.

Name	Statement	Parameters
Label	TRUNK	

The following control words are used in the TRUNK statement.

label

This required label specifies the name of the trunk. The label may be any name desired by the user that is from one to eight alphanumeric characters long. It must be unique from all other labels in the network configuration. Typical labels are ALPHA, BETA, etc.

TRUNK

This is the verb for this statement.

HOST Statement

The HOST statement provides NetEx with information about a particular host in the network. One HOST statement is required for each host in the network.

The HOST statement must follow the LOCALNET and TRUNK statements for that local network. All ADAPTER statements describing the configuration of the host must immediately follow the HOST statement. Note that the label is required. The parameters TYPE, MODEL, and OS are for clarity only and may be omitted. The parameters GROUP and PROTOCOL are used as needed and may be repeated within a single HOST statement

The HOST statement has the following format.

Name	Statement	Parameters
label	HOST	[TYPE = manufacture product line] [MODEL = model number] [OS = operating system name] [GROUP = group name] [PROTOCOL = n] [RATE = nnk] [OPTIONS = LONGMSG] [OPTIONS = ALTFIRST] [OPTIONS = NOAPR]

label

This required label specifies the logical name of the host. This label is to be the same name specified in the HNAME field of user connections. The label may be any name desired by the user which is from one to eight alphanumeric characters long. It must be unique from all other labels in the network configuration.

HOST

This is the verb for this statement.

TYPE

This optional parameter specifies the physical characteristics of the HOST by defining the trade name of the manufacturer's CPU product line. This parameter should be used to make the NCT more readable. The "manufacturer's product line" may be an alphanumeric string from one to eight characters long.

MODEL

This optional parameter specifies the model number within the manufacturer's product line. This parameter should be used to make the NCT more readable. The "model number" may be an alphanumeric string from one to eight characters long.

OS

This optional parameter specifies the operating system running on the machine. This parameter should be used to make the NCT more readable. The "operating system name" may be an alphanumeric string from one to eight characters long.

GROUP

This optional parameter specifies the logical name of a group of hosts that this HOST belongs to. If this HOST fits into more than one group, this parameter may be specified as many times as needed. The "group name" may be an alphanumeric string from one to eight characters long.

PROTOCOL

This optional parameter specifies the protocol level that will be used with this configuration. *n* may be specified as any decimal integer from 1 to 16. The number selected corresponds to the protocol level. When more than one level is acceptable, this parameter may be repeated, each time specifying a different level number. If a host is defined to belong to a GROUP, all the hosts in that group must have the same protocols selected.

Currently, this parameter must be specified as either PROTOCOL=2 or PROTOCOL=4. If it is not specified, PROTOCOL=2 is used as the default.

PROTOCOL=2 indicates that this host supports the NetEx/IP Type-2 protocol. This protocol results in more static usage of network bandwidth, and is usually preferred in local environments (LAN), or in short-distance configurations (e.g., < 500 miles, or less on paths with a high error rate).

PROTOCOL=4 indicates that this host supports the NetEx/IP Type-4 protocol. This protocol provides the ability for NetEx/IP to dynamically maximize the network performance, based on factors such as available bandwidth, distance, and workload on the network. To use Type-4 protocol on any given NetEx/IP connection, PROTOCOL=4 must be specified on the remote HOST definitions in the Configuration File, and optionally on the local HOST definition. If the local NetEx/IP supports Type-4 protocol, it will be used on connections to any remote hosts that have PROTOCOL=4 specified, regardless of how this parameter is specified, and also on any incoming Type-4 connections. Otherwise, Type-2 protocol is used.

RATE

This optional parameter specifies the limit on transmission rates to this host. The rate is expressed as a decimal number followed by K (kilobits per seconds).

OPTIONS

This optional parameter specifies options that apply to the host.

“NOAPR” indicates that the host does not support Alternate Path Retry.

“LONGMSG” forces a longer ‘message proper’ format that supports applications using high throughput application with Odata. LONGMSG = 1 must also be set in the initialization file.

“ALTFIRST” specifies that each new connect to this HOST will start with the next path in the path list after the last successful connect.

ADAPTER Statement

The ADAPTER statement describes each adapter and Bus Interface Unit (BIU) to NetEx. The ADAPTER statements for each adapter or BIU attached to a host must immediately follow the HOST statement.

Name	Statement	Parameters
[label]	ADAPTER	MODEL = nxxx NETADDR = xx SMGDREF = xx [CHANADDR = cuu] [NUMADDRS = n] [DEVNAME = device name] | T0 = label | | T1 = label | | T2 = label | | T3 = label |

label

This optional label specifies a symbolic name for the processor adapter or BIU. The label may be any name desired by the user which is from one to eight alphanumeric characters long. It must be unique from all other labels in the network configuration. It is helpful for operations if this is the same as the Control Unit name of the adapter.

ADAPTER

This is the verb for this statement.

MODEL

This required parameter defines the type of equipment attached to the HOST. Only processor adapters may be specified. The model number begins with an “A” or “N” (adapter) or a “B” (BIU), followed by three decimal digits.

NETADDR

This required parameter defines the hexadecimal network address of the adapter or BIU on the local network. The operand “xx” consists of two hexadecimal digits that specify the eight-bit adapter or BIU address. It must be unique from all other NETADDRs in this specific LOCALNET. This operand is required in all ADAPTER macros.

SMGDREF

This required parameter specifies the subaddress for this host’s session manager. The value corresponds to a specific subchannel or port for all input operations from the local adapter. It consists of two hexadecimal integer digits. The default value is “00”.

SMGDREF must equal the same value as the two low-order hex digits of CHANADDR (for example, if CHANADDR=240, then SMGDREF= 40).

CHANADDR begins a range of addresses; NUMADDRS identifies the number of addresses in this range. For example, if CHANADDR = 240, and NUMADDRS = 4, then the range of addresses contains 240, 241, 242, and 243. For Unisys, SMGDREF=40 (from the first address in the range - also the value of the two lower-order hex digits of the CHANADDRS). For IBM host-based NetEx, SMGDREF can be derived from any of the values in the range. (By convention, it is usually the highest value in the range.)

CHANADDR

This parameter, which is required only when using non-word channel adapters, specifies the lowest channel unit address of a group of units to be used by the NetEx software. This channel unit address must be expressed as three or four hexadecimal digits, for example, CHANADDR = 3C0.

Note: Unisys NetEx does not use this value, but it may be needed if a common NCT is used with an IBM system.

NUMADDRS

This parameter specifies the maximum number of adapter subaddresses that will be used by NetEx. The number of subaddresses must be expressed as a decimal number from 2 to 64 (example: NUMADDRS = 4).

DEVNAME

This parameter specifies a logical device name for this adapter. The "device name" may be from one to eight alphanumeric characters. The DEVNAME is required for Unisys Netex (for adapters on the local host). It identifies the device name which NetEx will use to assign the device. It must match the name assigned to the device in the OS2200 sysgen.

The other device names are generated by incrementing the last nonblank character of DEVNAME. It is suggested that the DEVNAME end in at least one "0" digit.

T0, T1, T2, and T3

This required parameter defines the trunks that are attached to the network adapter. At least one operand must be defined for each adapter. The associated labels specify the label of a preceding TRUNK statement. The referenced TRUNK must be defined in this LOCALNET.

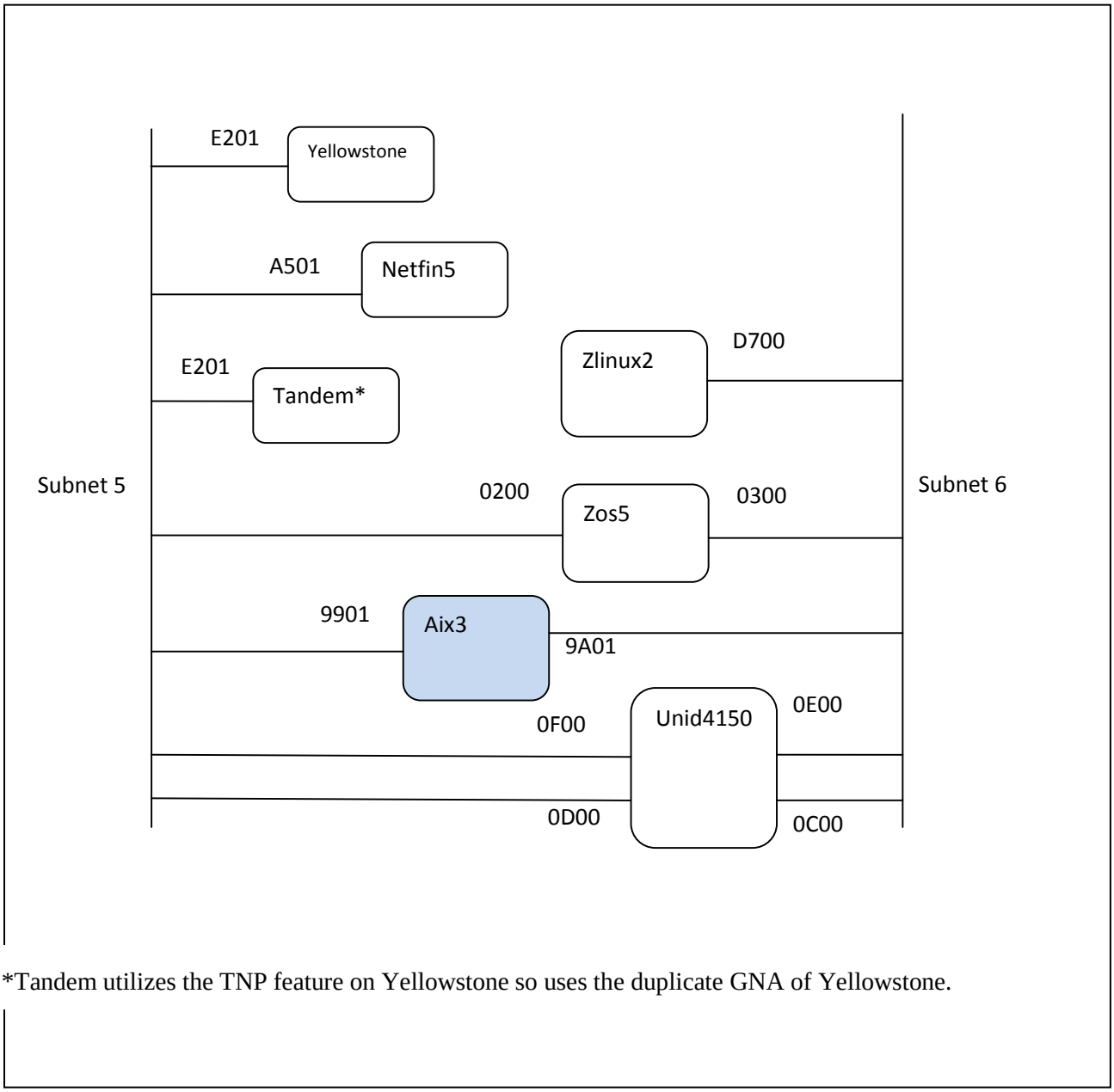
END Statement

The END statement indicates the end of the Network Configuration statements. This must be the last network configuration statement.

The END statement has the following format.

Name	Statement	Parameters
	END	

Network Configuration Example



*Tandem utilizes the TNP feature on Yellowstone so uses the duplicate GNA of Yellowstone.

Figure 12. Network Configuration Example

```
        VERSION    255
*****************************************************
NESINET  LOCALNET   TYPE=HC
UDP_ETH5 TRUNK
UDP_ETH6 TRUNK
*****************************************************
*       Host YELLOWST (SuSE Linux)
*****************************************************
YELLOWST HOST     TYPE=LINUX    MODEL=SuSE
                  PROTOCOL=2    PROTOCOL=4
         ADAPTER  MODEL=N130    NETADDR=E2
                  T0=UDP_ETH5   SMGDREF=01
*****************************************************
*       Host NETFIN5 (NESiGate-LO)
*****************************************************
NETFIN5  HOST     TYPE=NESiGate MODEL=X346
                  GROUP=MINGE  GROUP=FLASHE GROUP=TANDEMSE GROUP=DALE
                  PROTOCOL=2    PROTOCOL=4
         ADAPTER  MODEL=N130    NETADDR=A5
                  T0=UDP_ETH5   SMGDREF=01
*****************************************************
*       Requestor Hosts for use with H800IP/TNP
*  Tandem-s NonStop requestor connected to YELLOWSTONE
*****************************************************
TANYELL   HOST     TYPE=LINUX    MODEL=SuSE
                   PROTOCOL=2
          ADAPTER  MODEL=N130    NETADDR=E2
                   T0=UDP_ETH5   T1=UDP_ETH6
                   SMGDREF=01
*****************************************************
*  Host ZLINUX2
*****************************************************
ZLINUX2  HOST     TYPE=IBM      MODEL=Z9
                  PROTOCOL=2
         ADAPTER  MODEL=N130    NETADDR=D7
                  T1=UDP_ETH6
                  NUMADDRS=4    SMGDREF=00
*****************************************************
*       Host ZOS5 (Z/OS)
*****************************************************
ZOS5     HOST     TYPE=IBM      MODEL=Z9
                  PROTOCOL=2    PROTOCOL=4
         ADAPTER  MODEL=N220    NETADDR=02  CHANADDR=0200
                  T0=UDP_ETH5   NUMADDRS=4  SMGDREF=00
         ADAPTER  MODEL=N220    NETADDR=03  CHANADDR=0300
                  T0=UDP_ETH5   T1=UDP_ETH6 NUMADDRS=4    SMGDREF=00
*****************************************************
*       Host AIX3 (Power7)
*****************************************************
AIX3    HOST     TYPE=IBM_AIX  MODEL=POWER7
                 PROTOCOL=2
                 OPTIONS=LONGMSG ALTFIRST
        ADAPTER  MODEL=N130    NETADDR=99
                 T0=UDP_ETH5   SMGDREF=01
        ADAPTER  MODEL=N130    NETADDR=9A
                 T0=UDP_ETH6   SMGDREF=01
*****************************************************
*       Host Unisys
*****************************************************
UNID4150 HOST    TYPE=DORADO  MODEL=4150     GROUP=UNISYS
                 OPTIONS=ALTFIRST LONGMSG
        ADAPTER  MODEL=N220    NETADDR=0F    CHANADDR=0F00
                 T0=UDP_ETH5   NUMADDRS=4    SMGDREF=00
        ADAPTER  MODEL=N220    NETADDR=0E    CHANADDR=0E00
                 T0=UDP_ETH6   NUMADDRS=4    SMGDREF=00
        ADAPTER  MODEL=N220    NETADDR=0D    CHANADDR=0D00
                 T0=UDP_ETH5   NUMADDRS=4    SMGDREF=00
        ADAPTER  MODEL=N220    NETADDR=0C    CHANADDR=0C00
                 T0=UDP_ETH6   NUMADDRS=4    SMGDREF=00
         END
```

Figure 13. Network Configuration Statements

Operator Interface

The NetEx operator interface is designed to allow the computer operator to inspect the current status of NetEx and NetEx sessions, and to allow or deny access to NetEx resources such as local adapters, remote hosts, or particular types of NetEx service.

Entering NetEx Operator Commands

In batch mode, Unisys NetEx uses II$ to wait for operator input. When the II activity receives control, it displays the following message on the console via COMS.

```
'n - Enter NETEX command -->'
```

(In demand mode this message is displayed via ATREAD$.) When something is entered in response to the solicit, the input is processed and the solicit is re-displayed. To remove the solicit from the console, reply "END".

If the NetEx initialization statement 'CONSKY = xx' is specified when NetEx is started, the "II" operator keyin is no longer functional. No prompt appears, instead NetEx receives commands via ER KEYIN$. Each NetEx operator command must be prefixed with the value "xx" as noted in the CONSKY initialization statement. If "xx" is "NX", then to change MSGLVL, the operator enters:

```
NX SET MSGLVL 10
```

Any blank or punctuation character is considered to be a field separator. For example, the following are equivalent valid commands:

```
SET MSGLVL 10
set msglvl       10
set MSGLVL = 10
```

The following is an invalid command:

```
set MSGLVL10
```

Operator Commands

The following general command types are processed:

- DISPLAY commands give certain information concerning NetEx status.
- Starting and Stopping NetEx commands include KILL NETEX, ABORT NETEX, LOAD NCT, HALT (immediately terminate), DRAIN (gracefully terminate certain NETEX services), and START (restart drained or halted services).
- SET commands change NetEx parameters.

Remote Operator

NetEx provides a remote operator service that allows a user to request a NetEx operator display from other hosts on the network. This display may be requested from any NetEx that has the remote operator display feature enabled. The remote operator command is enabled using the NTXOPER initialization statement of the SET NTXOPER operator command. The class of commands that are available to the remote operator is either set with the NTXOPER or SET NTXOPER command, or may be set with the SET ROPCLASS operator command.

To use the remote operator feature, use the following command:

Command	Parameters
`>hostname[/applid]`	`NetEx-request`

>

This character indicates remote command

hostname

This is the remote host name as defined in the network configuration

/applid (optional)

This is the application ID on the remote host; it is optional. If requesting a remote NetEx operator display, this name is omitted. For an example, applid is MANAGER.

NetEx-command

This is the command to be executed by the remote program.

Examples:

```
>VM4381 DISPLAY PARMS
>VAX SHOW SESSION
>MVS4341/MANAGER DISPLAY STATS
```

The requested display will be shown in the format defined by the remote program. For example, if requesting display from an H267IP NetEx for HP OpenVMS, use an H260 command and receive an H260 display. (You will need to refer to the reference manual for the remote program.)

Remote Operator Command Classes

The remote operator service classifies commands into three classes in order to provide the desired level of authority to remote operators.

1. Class A commands severely affect NetEx. They include HALT, and DRAIN commands.
2. Class C commands are privileged commands. They include SET, START, and > (the remote operator command).
3. Class G commands are for status only. They include the DISPLAY commands.

DISPLAY Commands

The operator may display the current status of NetEx by using the NetEx DISPLAY commands. These commands permit the operator to interactively request displays from the operator's terminal or from an application program if the operator is allowed.

There are nine types of DISPLAY commands:

DISPLAY ADAPTER	Lists the adapters or units on an adapter.
DISPLAY DRIVER	Lists driver connections.
DISPLAY HALTED ADAPTERS	Lists all halted adapters.
DISPLAY HOST	Shows configured hosts.
DISPLAY IPROUTE	Shows GNA to IP mapping.
DISPLAY KEY	Shows the active key expiration date
DISPLAY MEMORY	Lists current buffer pool and memory usage.
DISPLAY NETWORK	Lists current active network connections
DISPLAY PARMS	Displays current parameter settings.
DISPLAY SESSION	Lists the sessions currently pending or in progress.
DISPLAY TRANSPORT	Lists the transport connections pending or in progress.
DISPLAY XMUSERS	Lists active users of cross-memory user blocks.

DISPLAY ADAPTERS Command

The DISPLAY ADAPTERS command displays the adapters and the local adapter configuration. It will also display the units and their status for a specific adapter.

Command	Parameters
`| DISPLAY |  | ADAPTER |` `| D       |  | A       |`	`| [adapter-name]    |` `| [network-address] |`

DISPLAY or D

This is a required keyword for this command.

ADAPTER or A

This is a required keyword for this command.

adapter-name

This optional parameter is a specific adapter name. It is the label on the ADAPTER configuration statement. This should also be the control unit name the Unisys operating system uses.

network-address

This optional parameter is a specific adapter network address in HEX. It is the network address of this adapter.

If no adapter is specified, the DISPLAY ADAPTERS display is similar to:

```
  Adapter # Type Unit State     Bytes/s  IN/OUT  Bytes/64s/s   I/O
  ------- ------ ---- -----     -------  ------  -----------  ----
  CUESE0  4 NGW1   A1  Running         0       0         0         0
  CUESF4  4 NGW1   E1  Running         0       0         0         0
  CUHYPB  4 N220   B4  Running         0       0         0         0
  CUESC1  4 N250   20  Halted          0       0         0         0
```

Figure 14. DISPLAY ADAPTERS output

Adapter

This is the control unit name of the adapter. It is the name that can be used on the "DISPLAY ADAPTER adapter-name" command.

Type

This is the model type of the adapter.

Unit

This is the HEX network address of this unit. This is the address that can be used on the "DISPLAY ADAPTER network-address" command.

State

This is specifies the current state of the adapter. The possible states are listed as follows:

- **Running** - is the normal state of the adapter.
- **Starting** - is in the process of starting up. This normally appears only if the unit is timing out.
- **Halting** - is in the process of halting the adapter. This is a long process. Each of the outstanding I/O requests must timeout and any Wait For Unsolicited Interrupt requests must timeout. This process takes about one minute.
- **Halted** - is the adapter is not currently in operation. This is the result of either excessive I/O errors or an operator request. Request a DISPLAY HALTED ADAPTERS to find the reason.
- **Draining** - drain has been set for the adapter. The adapter hardware is in the same state as running.

Bytes/s In/Out

This states the number of bytes received and sent on the adapter in the preceding second.

Bytes/64s I/O

This indicates the number of bytes/second rate received and sent (averaged from the previous 64 seconds).

The format for the "DISPLAY ADAPTER adapter-name" or "DISPLAY ADAPTER network-address" display is:

```
Adapter # Type Unit State     Bytes/s IN/OUT  Bytes/64s/s I/O
------- ------ ---- -----     --------------  ---------------
CUESF4  4 NGW1   E1  Running        0       0         0       0

Unit   #  Owner
------ -- ------
NES004 04 NETEX
NES005 05 OUTPUT
NES006 06 OUTPUT
NES007 07 OUTPUT
```

Figure 15. "DISPLAY ADAPTER n" output

Unit

This is the configured unit name.

#

This is the logical sub-address associated with this unit in HEX. It includes the logical unit and the channel address on the control unit. An address of 40 indicates the first unit on control unit 4.

Owner

This is the runid of the user who has this unit assigned. There are two special ids that are not really runids: NETEX and OUTPUT. "NETEX" is always the ID displayed for NetEx independent of the run's real ID; "OUTPUT" is used for a subchannel reserved for output.

DISPLAY DRIVER

The DISPLAY DRIVER command is designed to give information on the low level I/O activities within NetEx.

Command	Parameters
| DISPLAY | | DRIVER | | D | | D |	dref

DISPLAY or D

This is a required keyword for this command.

DRIVER or D

This is a required keyword for this command.

dref

This optional parameter is the specific driver reference number in hex (4-digits). If specified, only information for that dref is presented.

If dref=0, the statistics for all drefs are displayed and then cleared.

The output of DISPLAY DRIVER takes the following form:

```
 DREF User       Reads  Writes    R-data    W-data R-err W-err
 A100 *INIT*     10642       0   3300955         0     0     0
 A101 *OUT**         017376585         0  43618389     0     0
 A102 *OUT**         0       0         0         0     0     0
 A103 *OUT**         0       0         0         0     0     0
 E104 *INIT*   7613822       0  31399560         0     0     1
 E105 *OUT**         0  574511         0 176854221     0     0
 E106 *OUT**         0       2         0       232     0     0
 E107 *OUT**         0       0         0         0     0     0
```

Figure 16. DISPLAY DRIVER output

DREF

This specifies the unique reference number of this path to the driver. This value is significant in that it represents the hardware TO address that must be used to present data to this particular driver address from elsewhere in the network.

User

This is the runid of the user of the DREF.

Reads

This is the number of input message functions completed on this dref for this particular assign.

Writes

This is the number of output message functions completed on this dref for this particular assign.

R-data

This is the running total of input functions completed on this dref.

W-data

This is the running total of output functions completed on this dref.

R-err

This indicates the number of input failures due to hardware problems or trunk cancels.

W-err

This specifies the number of times that an output had to be retried due to transient errors or contention on the network.

DISPLAY HALTED ADAPTERS Command

The DISPLAY HALTED ADAPTERS command displays the local and remote adapters which have been halted.

Command	Parameters
| DISPLAY | | HALTED | | ADAPTERS | | D | | HA | | A |	

DISPLAY or D

This is a required keyword for this command.

HALTED or HA

This is a required keyword for this command.

ADAPTERS or A

This is a required keyword for this command.

The DISPLAY HALTED ADAPTERS display is:

```
Adapter   Halt Type Proximity
--------  --------- ---------
CUESC1    IO ERROR  LOCAL
```

Figure 17. DISPLAY HALTED ADAPTERS output

Adapter

This is the control unit name of the adapter or its NCT network address in hex. Local adapters are displayed by name unless the name is not present on the NCT ADAPTER statement and remote adapters by network address.

Halt Type

This indicates why this unit is halted. OPERATOR indicates that the operator HALTED this adapter. I/O ERROR indicates that a fatal condition, such as an I/O status of DOWN, has occurred.

Proximity

This indicates whether the adapter is LOCAL or NON-LOCAL.

DISPLAY HOST Command

The DISPLAY HOST command gives the operator a list of the hosts defined on the network. By specifying a host name, the operator may limit the display to only the specified host and receive more detailed information. An * preceding the hostname indicates that the host is in drained status, and cannot be used.

Command	Parameters
| DISPLAY | | HOST | | D | | H |	[hostname]

DISPLAY or D

This is a required keyword for this command.

HOST or H

This is a required keyword for this command.

hostname

This optional parameter is the name of the host to be displayed. By omitting this parameter, information about all hosts will be displayed.

If no host name is specified, a display similar to the following will appear:

```
Host UNISYS   has routes to the following HYPERCHANNEL hosts:
-------------------------------------------------------------
NTXLCL05 NTXLCL04 NTXLCL03 NTXLCL02 NTXLCL01 NTXLCL00 OS390F   OS390E
OS390H   ZARKHOVH ULTRA5E  ULTRA5F  ULTRA5H  TANDEMSE SOLSRVRE SOLSRVR
SOLSRVRH RIOS4E   RIOS4F   RIOS4H   NETFIN1E NETFIN1F HPE      HPF
HPH      DXUB5H   DXU20H   NTXLCL

Host UNISYS   has an INTRA-HOST route to itself.
```

Figure 18. DISPLAY HOST output

If a host is specified, the display is:

```
Host UNISYS   has the following routes to host HPH :

 PATH MAX SIZE  MAX RATE  PATH      ADDRESSES     TRUNK    LINK ADDRS
 NUM  BYTES     BITS/SEC  DELAY     LOCAL REMOTE  MASK     LOCAL REMOTE
 ---  -------   -------   ------    ----  ----    ----     ----  ----
   1  65.000K   136.00M      1MS    2000  DC00    88
   2  65.000K   136.00M      1MS    B4B0  DC00    88
```

Figure 19. DISPLAY HOST n output

DISPLAY IPROUTE Command

The DISPLAY IPROUTE command displays the current GNA to IP address mapping.

Command	Parameters
| DISPLAY | | IPROUTE | | D | | IP |	[gna]

DISPLAY or D

This is a required keyword for this command.

IPROUTE or IP

This is a required keyword for this command.

gna

(optional) specific GNA to display

The DISPLAY IPROUTE output is similar to:

```
  GNA   IPaddress         GNA   IPaddress         GNA   IPaddress
0A00 10.1.5.12 D        0B00 10.1.5.157 D       0C00 10.1.6.18 LD
0C20 10.1.6.19 D        0D00 10.1.5.16 LD       0D20 10.1.5.19 D
0E00 10.1.6.18 LD       0E20 10.1.6.19 D        0F00 10.1.5.16 LD
0F20 10.1.5.19 D        1801 10.1.5.170 D       3100 10.1.5.15 D
GNA-IP total entry count = 12
```

Figure 20. DISPLAY KEY output

GNA are separate representations of a GNA address (must also be in the NCT)

IPaddress is the associated IP address for this GNA (see 'SET IP' command)

'D' (after the ipaddress) indicates the mapping was done by DNS lookup

'L' (after the ipaddress) indicates a local adapter GNA

DISPLAY KEY Command

The DISPLAY KEY command displays the current key expiration date.

Command	Parameters
| DISPLAY | | KEY | | D | | K |	n/a

DISPLAY or D

This is a required keyword for this command.

KEY or K

This is a required keyword for this command.

The DISPLAY KEY output is:

```
 H300IPC key expires on: yyyy-mm-dd
 H300IPC last operational date: yyyy-mm-dd
```

Figure 21. DISPLAY KEY output

The key expires on shows your license expiration date.

The last operational date shows when the product will cease to function.

DISPLAY MEMORY Command

The DISPLAY MEMORY command displays the current status of the NetEx buffer pools or the actual contents of NetEx memory addresses.

The DISPLAY MEMORY command has the following format.

Command	Parameters
| DISPLAY | | MEMORY | | D | | M |	address

DISPLAY or D
This is a required keyword for this command.

MEMORY or M
This is a required keyword for this command.

address
This is the octal (leading zero) or decimal NetEx address whose contents are desired. Four memory locations are displayed.

The DISPLAY MEMORY output is:

```
            Buffer Pools                Common Data Banks (NTX$DBn)
 Size  Avail  Min   Max   Avg  Que  Bank Avail   Min    Max   Que
 ----  ----- ----- ----- ----- ---  ---- ------ ------ ------ ---
   16     60    55    60    59   0     0  53704  16872  65496   0
   32     90    84    90    89   0     1  65496  65496  65496   0
   64    158   144   180   158   0     2  65496  65496  65496   0
  128     13     7    15    12   0
```

Figure 22. DISPLAY MEMORY output

Buffer Pools describes the data area that NetEx maintains for its own internal use.

- **Size** The size of the buffers in this pool
- **Avail** Number of buffers of this size currently available
- **Min** Minimum number of these buffers that was ever available
- **Max** Maximum number of these buffers that were ever available
- **Avg** Average number of these buffers that are available
- **Queue** Number of requests that are currently waiting for buffers of this size

Data Areas describes the data areas used to hold user data while it is being sent and waiting to be read. There is normally only one bank (bank 0). If the NXDEFS tag DBCNT is changed, there will be one line displayed for each common data bank.

- **Bank** Bank number, starting with bank 0
- **Avail** Number of words currently available in this bank
- **Min** Minimum number of words ever available in this bank
- **Max** Maximum number of words ever available in this bank

Queue Number of requests that are currently waiting for buffers

DISPLAY NETWORK

The DISPLAY NETWORK command gives the operator a list of the network layer connections that are currently in progress.

Command	Parameters
| DISPLAY | | NETWORK | | D | | N |	

DISPLAY or D

This is a required keyword for this command.

NETWORK or N

This is a required keyword for this command.

```
   Nref User    State    L-Drefs-R Trks MaxPData MaxRate Delay
      0 BFXJS   idle     0000 0000  00         0       0     0
     36 *OPIF*  data     0600 0600  88    524288   80000     1
     38 *RREC*  data     0600 0600  88    524288   80000     1
  65535 *SMGR*  offered  0000 0000  00         0       0     0
```

Figure 23. DISPLAY SESSION output

NREF

This is the unique identifier that distinguishes this connection from all other active connections to this NetEx. It is the same as the corresponding SREF and TREF.

User

This is the runid of the job that requested the service.

State

This is the current state of the connection.

L-Drefs-R

The local and remote DREFs (GNAs) for this connection.

Trks

The path trunkmask (obsolete).

MaxPdata

This is the maximum segment size for writes on this path (in bits)

MaxRate

This is the maximum send rate for this connection (in Kbits/sec)

Delay

This is the configured round-trip delay for this path (in msecs)

DISPLAY PARMS

The DISPLAY PARMS command displays most of the parameter values controlled by the set command and some NetEx conditions.

Command	Parameters
| DISPLAY | | PARMS | | D | | P |	

DISPLAY or D

This is a required keyword for this command.

PARMS or P

This is a required keyword for this command.

This command causes a display similar to:

```
    H300IPC  7.1   parameters for host UNISYS
 ------------------------------------------------------------
 Sesmax=15       WdogInt=2      MsgLvl=0       RTlvl=27    MaxSeg=2048
 Maxbo=20000     Maxbi=20000    Defbo=8191     Defbi=8191
 ConTO=12        DeadTO=30      IdleTO=5       ReadTO=30
 Tracing is  ON  MEMORY  10x1400-word buffers  NTXOPER=3/A
 Tracing Events: DA SH OO OI BF BA UT UD UB UC UR IC IR DO DI MR MC
 Status: NORMAL  NCT: 1  SUP-rate: 0  NtxNum: 0  IPClog/cons: 5/*
 NTX$:020000/0243647  COMN:0403/010116  DBx:0404/0300000/0744760/1
 Created: APRIL 29, 2010 08:53:42   Started: APRIL 29, 2010 08:58:52
```

Figure 24. DISPLAY PARMS output

Sesmax

This is the number of session connections permitted at one time. This parameter may be changed using the SET SESMAX command.

WdogInt

This is the number of seconds that the watchdog timer waits before checking the NRB timeout values. This parameter may be changed using the SET WDOGINT command.

MsgLvl

This is the minimum level of severity of messages that are to be displayed to the operator. All messages of greater than or equal priority will be displayed. This parameter may be changed using the SET MSGLVL command.

Message Level	Description
15	Fatal errors. Immediately after this message, NetEx will stop.
14	Responses to operator-initiated actions.
12	Non-transient adapter I/O errors
8	High priority messages.
7	Session related messages; e.g., connect failures, etc.
6	Dropped messages. Transient errors.
4	Session starts and ends.
0	Debug diagnostics. This currently displays any bad status on a driver level return.

RTlvl

This is the real time switching level of the NetEx program main activity as set by the Initialization Parameters. This parameter may not be changed.

MaxSeg

The segment size in words that the message and data will be broken into for transporting across the network.

Maxbo

This is the maximum buffer output size (in words) that a user may specify for data going from this host in a single message. This parameter may be changed using the SET MAXBO command.

Defbo

This is the default buffer output size (in words) that a user may specify for data going from this host in a single message. This parameter may be changed using the SET DEFBO command.

Defbi

This is the default buffer input size (in words) that a user may specify for data coming to this host in a single message. This parameter may be changed using the SET DEEBI command.

ConTO

This is the maximum number of seconds that NetEx will wait for a transport connect message or the first message of a re-path attempt to generate a response from a remote host. After this time, a new path will be tried. This parameter may be changed using the SET CONTO command.

DeadTO

This is the maximum number of seconds that NetEx will wait for an active connect to receive a response from the remote host. After this time, a new path will be attempted. This parameter may be changed using the SET DEADTO command.

IdleTO

This is the maximum number of seconds that NetEx will wait with no activity on a connection before sending an idle message to verify the continued existence of a party at the other end of the logical connection. This parameter may be changed using the SET IDLETO command.

ReadTO

This is the number of seconds that NetEx will retain user data while waiting for the receiver to issue a read request. This parameter may be changed using the SET READTO command.

Tracing

This specifies the trace options selected. The trace may be ON or OFF; it may be going to MEMORY, DISK or TAPE. These parameters may be changed using the SET TRACE command.

NTXOPER

There are two fields displayed. The first is the maximum number of active remote operator receivers. The second is the privilege level of the remote receivers (A, C, or G). The number of remote operator receivers may be changed using the SET NTXOPER command. The privilege level of the remote operators may be changed with the SET NTXOPER or the SET ROPCLASS commands.

Tracing Events

This specifies the events currently being traced (if Trace is ON). NetEx may be tracing ALL or selected events. Table 8 lists the possible trace events. These parameters may be changed using the SET TRACE command.

Status

This is the current status of NetEx, NORMAL or DRAINING. The status may be set using the DRAIN NETEX or START NETEX commands.

NCT

This is the value given in the NCT VERSION statement.

SUP-rate

This refers to the OS2200 Standard Units of Processing (SUPs). It is the average total SUPs/second charged to NetEx over the past eight seconds.

NtxNum

Index assigned to this Netex instance (0-6), Set in initialization parameters.

IPClog/cons

The level of logging & console messages being issued by the IPC component. See the SET IPCLOG & SET IPCLONS commands for values.
'*' indicates that the IPC program default value is in effect (has not been changed by Netex.)

NTX$:

BDI, start and end addresses of the Netex program bank

COMN:

BDI and end addresses of the Netex common I-bank

DBn:

BDI, start, end addresses, and number of the Netex common D-banks

Created & Started:

Date & time of the program build and start of execution.

Table 5. Trace Events

Trace Event	Description
ALL	All events listed in this table
BA	Buffer Allocation
BF	Buffer Free
DI	Driver Input (message proper only)
DO	Driver Output (message proper only)
DA	Driver Associated Data
IC	I/O Completion
IR	I/O Request
MC	Module Call
MR	Module Request
OI	Operator command Input
OO	Operator Output
QM	Queue Manager

Table 5. Trace Events	
Trace Event	**Description**
SP	Spawn - start a new NetEx task
SH	Trash – discarded messages from the network
SU	Suspend - queue a NetEx task
UB	User Buffer copied
UC	User Request Cleanup
UD	User Request Done
UR	User Request initiation
UT	User program Terminated

DISPLAY SESSION

The DISPLAY SESSION command gives the operator a list of the sessions that are currently in progress. A session is defined as being in progress between the times the user issues an SOFFER or SCONNECT to the time a Disconnect Indication is received by the user.

Command	Parameters
| DISPLAY | | SESSION | | D | | S |	sref

DISPLAY or D

This is a required keyword for this command.

SESSION or S

This is a required keyword for this command.

sref (optional)

This is the specific session reference number. If specified, only information for that session is presented. If not specified, information for all srefs is presented, in a tabular format:

```
NREF  User   Name     State    DestHost,Nref  RmUser  BlksOut BlksIn Otime
 7489 JRSA   USER     data     OS390H       5 NUANSERV      9     11
 7491 JRSA   USER002  data     OS390H       7 SIGNOJR       7    326
```

Figure 25. DISPLAY SESSION output

NREF

This is the unique identifier that distinguishes this session from all other active connections to this NetEx. This reference identifier must be used for operator commands that modify a session, such as session halt.

User

This is the runid of the job that requested the service.

Name

This is the identifier that the program issued to allow the connection to take place. It is the name that must be supplied by the NetEx user whenever an SCONNECT or SOFFER takes place.

State

This gives an indication of the current condition of the session connection. The possible states for a session connection are:

- **unasgd** indicates the session is idle. This is normally only as it is being established or termination. This state should not persist for more than a few seconds.
- **offered** indicates that the application program has "advertised" the resource listed under "NAME" to NetEx. The application program is waiting for some other entity in the network to issue a SCONNECT to match with the offered NAME.
- **connout** indicates that a SCONNECT has been sent by this user to the remote host, and that the user is expecting a Connect Confirmation from the remote application.
- **confirm** indicates that an OFFERed connection has received a connection, and that NetEx is waiting for the application to respond to the connecting application.
- **data** indicates that the connect process has completed, and that normal data transfer is taking place between the two parties.
- **disconn** indicates that a Disconnect has been initiated from either the local or remote host, and NetEx is waiting for the user to read the Disconnect Indication.
- **closout** indicates this session has issued a CLOSE.
- **closin** indicates this session has received a CLOSE indication from the remote application.
- **closed** indicates this session has sent and received a CLOSE. The connection is waiting for all data to be acknowledged. This state may exist for a long period, depending on CONTO and DEADTO.

DestHost/NREF

This is the name of the corresponding host for the session, and it's NREF on the remote host.

RmUser

This is the runid of the application on the remote host.

Otime

This is the offer timeout value from NRBTIME.

BlksOut/In

This is the number of application blocks sent/received for this session.

DISPLAY TRANSPORT

The DISPLAY TRANSPORT command is designed to give the operator an indication of the transport connections taking place at the current time. This provides information on the users external to NetEx that are using Transport services directly, as well as more detailed information on the progress of users who are using session services. The Path information is displayed only if this side made the connection. If this side put up the offer, the path fields will be blank, as the other side made all the path decisions.

Command	Parameters
| DISPLAY | | TRANSPORT | | D | | T |	tref

DISPLAY or D

This is a required keyword for this command.

TRANSPORT or T

This is a required keyword for this command.

tref

This optional parameter is the specific transport reference number. If specified, detailed information for that tref is presented. If not specified, general information for all trefs is presented.

The format of DISPLAY TRANSPORT output if no TREF is specified.

```
 NREF Remote User     BLKO  BLKI Segsz State    T-out   T-in  Paths
 7489      5 JRSA     4096  4096  2000 data         9     11   1/1/0
 7491      7 JRSA     4096  4096  2000 data         7  34152   1/1/0
65535      0 *SMGR*   8191  8191  2000 offered      0      0
```

Figure 26. DISPLAY TRANSPORT output

NREF/Remote

These are the local and remote transport reference numbers. An NREF is a unique identifier assigned to this transport connection that distinguishes it from all other connections at the current time.

User

This is the runid or internal NetEx id of the party that issued the transport connection,

BLKO

This identifies the maximum output block size which may be sent over this connection.

BLKI

This specifies the maximum input block size which may be received by this connection.

State

This is the current status of the transport connection. This information is often useful in determining if a connection is "hung" during the starting or ending phases of a connection. The possible states, with a brief explanation of each, are as follows.

- **unasgd** indicates the transport connection is idle. This is the state of the transport connection when an offer is outstanding and no connect has occurred. This is also the state for a brief period at termination.
- **offered** indicates the transport connection is available for a connection.
- **confirm** indicates that a connect has arrived matching a previous offer. The application is expected to issue a confirm or disconnect at this point.

- **connout** indicates that a connect has been requested and is in the process of being sent.
- **consent** indicates that a connect has been sent. This does not imply that the other end has acknowledged receipt of the connect,
- **confin** indicates that a confirm has been received from the other end, but has not been read by the connecting application. A READ should be issued to input the confirm.
- **data** indicates that the connection is fully established and that normal data may be exchanged with SREAD and SWAIT (or the transport equivalents).
- **disconn** indicates that a disconnect sequence is in progress. Normally this occurs very quickly. If this state persists, look for I/O errors or a NetEx problem.
- **closout** indicates this session has issued a CLOSE.
- **closin** indicates this session has received a CLOSE indication from the remote application.
- **closed** indicates this session has issued and received a CLOSE. The connection is now waiting for all data to be acknowledged. This state remains in effect for many seconds or a few minutes, depending on DEADTO and CONTO.
- **waiting** indicates that all of the CLOSE-related operations have completed on this side. NetEx is waiting for DEADTO number of seconds to see if the other side is finished.

T-out / T-in

These indicate the number of messages that have been sent and received for this transport connection.

Prot

Transport protocol type for this connection.

Paths (C/L/B)

This describes the route information for this connection. C is the current route being used by NetEx, L is the last route data has been sent across on, and the B is the best route available. (This field is currently not meaningful.). This information does not appear for the offering side.

If a specific TREF is specified on the DISPLAY TRANSPORT command, detailed information concerning only that t-connection is displayed:

```
  NREF Remote User     BLKO  BLKI Segsz State    T-out   T-in  Prot Paths
  7491      7 JRSA     4096  4096  2000 data         7  56804    2  1/1/0
  Rate Delay Buffers  LRNA  PBNA  LRNU Rexmits Prot Wm/Tr/Ak/Dt/Rd-Qs
100101     1   2   0  1117  1116     2       0    2  0 0 0 0 1
NTXbuf  Srate Wait  RTms/RTmin/RTmax    Pipe InPipe Rate%Equv
202720  12512    0     2     1   117   23128      0  100  850
        Rcvd  Sent AckBits T/O Credit Proc/Rrate
local: 56806     7 000000   72     2    56807
remote:    7 56806 000000          2       11
```

Figure 27. DISPLAY TRANSPORT n output

The first line is described in the previous figure. The rest of the information displayed is described below:

Rate

This is the maximum transmission rate (in Kbits/second).

Delay

This is the maximum path delay in milliseconds.

Buffers

This is the maximum and active number of transmit buffers.

LRNA
This is the next Logical Record Number to be assigned for this connection.

PBNA
This is the next Physical Block Number to be assigned for this connection.

LRNU
This is the next LRN to be given to the user.

Rexmits
This is the number of segments retransmitted due to NAKs.

Wm/Tr/Ak/Dt/Rd-Qs
These are the number of NITs queued: waiting for memory /for transmission/ for acknowledgement/ data waiting for read/ reads waiting for data.

NTXbuf
This is the address of the NetEx Control Block for this session

Srate
This is the current send rate in KB/s.

Wait
This is the calculated wait time required between outputs to maintain the correct rate in msecs.

RTms/Rtmin/Rtmax
These are the current/minimum/maximum transmit delay (msecs) for the past measuring interval.

Pipe
The number in words allowed to be unacknowledged (in the pipe) - usually 2*rate*delay.

InPipe
The number in words currently in the pipe (protocol 4 only). (Protocol 4 is currently not supported.)

Rate%Equv
Calculation percentages used to adjust protocol 4 send rate. (Protocol 4 is currently not supported.)

Rcvd
This is the highest PBN received.

Sent
This is the highest PBN transmitted.

Ackbits
This is the ACK/NAK bit-significant field read or sent. A bit = 1 indicates a NAK'd block.

T/O
This is the transmit timeout (how long the session will remain active with no communication).

Credit
This is the ACK credit (number of PBNs allowed before acknowledgement is required).

Proc/Rrate
This is the highest LRN that may be transmitted (protocol 2 proceed), or, the latest receive rate reported by the other transport (protocol 4). (Protocol 4 is currently not supported.)

local:
The information presented in this row is for the local host.

remote:
The information presented in this row is for the remote host.

DISPLAY XMUSERS Command

The command DISPLAY XMUSERS or D X shows the currently active cross-memory users.

Command	Parameters
| DISPLAY | | XMUSERS | | D | | X |	*

If the optional parameter "*" is omitted, the response is as follows:

```
XMUBs total active count = n
```

Where *n* is the number of currently active XMUBs.

If the optional parameter "*" is present, the response is as follows (where each line represents one active XMUB):

```
  XMUB   XMuser state    Sn#  Rq# UsrBDI NtxBDI
  003716 EAT    assigned   1    1 000005 000005
```

where

XMUB
The address of the user block in NTX$COMN

XMuser
The runid of the user

State
The current state of the XMUB, which can be "assigned", "terming", or "termed"

Sn#
The number of sessions for this user

Rq#
The outstanding requests for this user

UsrBDI
The bank index of user Dbank

NtxBDI
The bank index of Netex Dbank

Starting and Stopping NetEx Resources

DRAIN ADAPTER Command

The DRAIN ADAPTER command is designed to begin the process of removing a locally attached network adapter from service. No sessions that are currently in progress are affected; they will continue to use their current adapter address assignments. Subsequent NetEx users establishing a connection will have a locally attached adapter that is not drained assigned to them. If an application requests a specific adapter address, or all adapters attached to a host are drained, then the connection will be rejected with an error.

The DEVNAME associated with the DRAIN ADAPTER command is the device name of the adapter as configured in the NCT.

Command	Parameters
| DRAIN | | ADAPTER | | P | | A |	devname

DRAIN or P
This is a required keyword for this command.

ADAPTER or A
This is a required keyword for this command.

devname
This required parameter is the NetEx name of the adapter being drained.

DRAIN NETEX Command

The DRAIN NETEX command is used to begin an orderly shutdown of the NetEx program. No connections that are currently in progress will be affected. Any new attempt to offer a service, establish a connection, or assign a driver path will be rejected with an error code. When an application terminates a connection, it will not be permitted to establish a new one.

Command	Parameter
| DRAIN | NETEX | P |	

DRAIN HOST Command

The DRAIN HOST command is used to temporarily block new access to specific hosts or to a specific path to a host.

Command	Parameters
| DRAIN | | HOST | | P | | H |	hostname [PATH n]

DRAIN or P
This is a required keyword for this command.

HOST or H
This is a required keyword for this command.

host name
This required parameter is the name of the host to be drained.

PATH n
This optional parameter indicates a specific path to the host which is to be drained. The n indicates the path number to be DRAINed (l...#paths).

HALT ADAPTER Command

The HALT ADAPTER command can be used to discontinue use of any network path containing the HALTed adapter. This command should only be used if there is another adapter available for NetEx's use. This command will affect only those sessions whose routes must use the adapter that is being halted. The halt process may take up to thirty seconds for a locally attached adapter. The device(s) are FREEd from NetEx and are available for assignment (by a diagnostic program, for example). The response from NetEx is:

```
ADAPTER  devname HAS BEEN HALTED
```

The partitioning of a multi-processor 2200 System, along with an IOP connected to an active adapter, will cause NetEx to HALT that adapter automatically. The response will be the following:

```
ADAPTER devname HAS BEEN DOWNED
```

The devname associated with the HALT ADAPTER command is the cuname of the adapter as configured in the NetEx NCT.

Command	Parameters
| HALT | | ADAPTER | | H | | A |	| devname | | unitnum |

HALT or H
This is a required keyword for this command.

ADAPTER or A
This is a required keyword for this command.

devname
This is the cuname of the local adapter being HALTED.

unitnum
This is the hex unit number of the local or remote adapter being halted.

START ADAPTER Command

The START ADAPTER command permits access to a local adapter interface. It reverses the effect of a previous DRAIN ADAPTER or HALT ADAPTER command. This command cannot be used to introduce a new adapter into the configuration,

Command	Parameters
| START | | ADAPTER | | S | | A |	| devname | | unitnum |

START or S
This is a required keyword for this command.

ADAPTER or A
This is a required keyword for this command.

devname
This is the NetEx name of the local adapter to be started.

unitnum
This is the two digit hex unit number of the local or remote adapter being started.

START NETEX Command

The START NETEX command reverses the effect of a previous DRAIN NETEX command. Applications can once again initiate NetEx connections.

Command	Parameters
| START | | NETEX | | S | | N |	

START or S
This is a required keyword for this command.

NETEX or S
This is a required keyword for this command.

START HOST Command

The START HOST is used to start specific hosts that were previously drained.

Command	Parameters
| START | | HOST | | S | | H |	hostname [PATH n]

START or S
This is a required keyword for this command.

HOST or H
This is a required keyword for this command.

hostname

A required parameter is the name of the host to be started.

PATH n

(optional) This indicates a specific path to the host which is to be started. The n indicates the path number to be started.

Miscellaneous NetEx Commands

ABORT Command

The operator can immediately stop NetEx using the ABORT command. Like the KILL command, NetEx activity is immediately ended. Unlike the KILL command, PMD and dump analysis are inhibited. This command is not available through the remote operator interface.

Command	Parameters
`ABORT [NETEX]`	

ABORT
This is the keyword for this command.

NETEX
This is an optional keyword for this command.

CLEAR IPROUTE Command

The CLEAR IPROUTE command removes the association between a GNA address and an IP address.

Command	Parameters
`CLEAR [ IPROUTE | IP ]`	`Gna`

CLEAR
This is a keyword for this command.

IP or IPROUTE
This is a keyword for this command.

Gna
This is a local or remote GNA address (4 hex digits)

KILL Command

The operator can immediately stop NetEx using the KILL command. This command immediately terminates all NetEx activity. Connections in progress are terminated and a 0512 return code is inserted into all active NRBs. This command is not available through the remote operator interface.

Command	Parameters
`KILL [NETEX]`	

KILL
This is a required keyword for this command.

NETEX
This is an optional keyword for this command.

LOAD KEY Command

At any time, the operator may cause the reload of the license key from KEY. After the LOAD a DISPLAY KEY is performed automatically. If the expiration date for the new key is not later than the current one then the new key is not used.

Command	Parameters
LOAD KEY	

LOAD
This is a keyword for this command.

KEY
This is a keyword for this command.

LOAD NCT Command

At any time, the operator may cause the reload of NetEx configuration path information. If the network configuration changes for any reason and the Configuration Manager is run to produce a new output file, then "LOAD NCT" loads the new file for use on subsequent connections or re-pathing. The omnibus element in file NETEX$CONFIG.hostname is loaded. This does not affect current sessions in progress.

If you are configured to use DNS for GNA to IP address mappings, Netex will issue a request to IPC to retrieve all the current mappings. If an IP address changes, this could affect a connection that is currently in progress.

This command will not change the local adapter configuration – a Netex restart is required to add or delete a local adapter.

Command	Parameters
LOAD NCT	

LOAD
This is a keyword for this command.

NCT
This is a keyword for this command.

MSG Command

The MSG command allows a message to be sent via the NetEx Remote Operator to be displayed at the remote NetEx console or terminal. (Available only in Unisys Netex.)

Command	Parameters
>hostname MSG	<message text>

hostname
This is the remote host name as defined in the network configuration.

MSG
This is a required keyword for this command.

message text
This is the message to be displayed at the remote NetEx console.

SWITCH Command

The switch command is valid for Netex and IPC. It closes the current breakpointed print dataset on disk and creates a new cycle. All future print will go to the new cycle. The LOGQ, LOGF and USE parameters must be configured before it is available to be used by Netex. The IPCUSE parameter must be configured before it is available to IPC.

Command	**Parameters**
SWITCH	

This command requires the NTX$INIT file to contain the USE, LOGQ, LOGF and IPCUSE parameters. To execute this command shortly after midnight, you may use the automation tools on the OS 2200 system. The rule would be similar to:

```
DEFINE "NETEXOAN" 2
  MESSAGE “H300IPC 7.4.2-6YY (nnnnnn nnnn:nn) 2013 Oct 23 Wed nnnn:nn"
  TYPE ANY-SENDER
  PRIORITY 128
  INSTANCE PRIMARY
  RUNID “NTX0”
  TOKEN FIXED 1 "H300IPC"
  TOKEN MASKED 2 "7.4\*7\"
  TOKEN MASKED  9 "0000:\*2\"
  ACTION ALL EXECUTE "SWITCH"
  END
```

NTX0 would be replaced by your runid. You would also have to add the macro SWITCH to the automation software. The macro would specify your “*netex_cmd_keyin* SWITCH”, “*ipc_cmd_keyin* SWITCH”, along with any additional commands you may wish to execute.

Miscellaneous IPC Commands – USE IPCKEYIN

ABORT Command

The operator can immediately stop IPC and all copies of NETEX using this copy of IPC.

Command	Parameters
ABORT	

ABORT
This is the keyword for this command.

KILL Command

The operator can immediately stop IPC and all copies of NETEX using this copy of IPC.

Command	Parameters
KILL	NTXNUM

KILL
This is the keyword for this command.

NTXNUM is the netex number that will be terminated. The valid numbers are 0 through 5. The operator should verify that the netex is down, before using this command. If this netex is still active, it will be terminated.

SWITCH Command

See the discussion under Miscellaneous NetEx Commands.

TERM Command

The operator can request a graceful termination of IPC. All copies of NETEX using this IPC was first be terminated. A warning message is produced if a NETEX is still active. It will display the NTXNUM of the active netex.

Command	Parameters
TERM	

Term
This is the keyword for this command.

Setting NetEx Parameters

The SET commands are used to modify particular events or settings within NetEx. The following \ SET commands are currently supported (others are for protocol 4 which is not supported at this time):

SET CONTO specifies the maximum number of seconds that NetEx will wait for a transport connect message to generate a response from the remote host.

SET DEADTO specifies the maximum number of seconds that NetEx will wait before it disconnects a transport connection because there was no response from a remote host.

SET DEFBI specifies the default buffer input size (in bytes) that a user may specify for data coming in to this host in a single message.

SET DEFBO specifies the default buffer output size (in bytes) that a user may specify for data going out from this host in a single message.

SET DRIVMAX specifies the maximum number of concurrent Driver Interface users supported by NetEx.

SET IDLETO specifies the number of seconds that NetEx transport will wait for before sending an idle message to verify the continued existence of a party at the other end of a logical connection.

SET IPROUTE maps NCT GNA addresses to network IP addresses.

SET IPCONS specifies the console message level for the IPC component.

SET IPCLOG specifies the logging message level for the IPC component.

SET MAXBI specifies the maximum buffer input size (in bytes) that a user may specify for data coming in to this host in a single message.

SET MAXBO specifies the maximum buffer output size (in bytes) that a user may specify for data going out from this host in a single message.

SET MAXOD specifies maximum user octet data allowed.

SET MSGLVL specifies the minimum level of severity of messages that are to be displayed to the operator.

SET NTXOPER specifies if the remote operator service is to be enabled or disabled, and the number of remote operator sessions to allow.

SET PATH limits adapter paths for testing purposes.

SET PRINT controls the printing of messages into the NetEx log

SET ROPCLASS specifies the class of operator commands that remote operators will be allowed to issue.

SET READTO specifies the number of seconds that NetEx transport will retain user data while waiting for the receiver to issue a read request.

SET SESMAX specifies the number of session connections or OFFERs permitted at one time.

SET TRACE specifies which trace events are to be saved and where they are to be stored.

SET WDOGINT specifies the watchdog timeout value to be used when timing NetEx events.

Note: In all of the following SET commands, the equal sign is optional.

SET CONTO Command

The SET CONTO command specifies the maximum number of seconds that NetEx will wait for a transport connect message or the first message on a new path to generate a response from the destination host. If this time is exceeded, the transport will assume the path is "down" and will attempt to re-path to a new path. The transport connect message is re-sent every IDLETO seconds until CONTO seconds have passed.

Command	Parameters
`SET CONTO`	`= seconds`

SET
This is a keyword for this command.

CONTO
This is a keyword for this command.

seconds
This specifies the number of seconds that NetEx will wait for a transport connect message to generate a response from the destination host.

SET DEADTO Command

The SET DEADTO command specifies the amount of time transport will wait until it disconnects a session because there was no response from the remote host. The remote host normally generates an idle message every IDLETO seconds based on its own IDLETO parameter. Receipt of any message from the remote host keeps the DEADTO timer from expiring.

Command	Parameters
`SET DEADTO`	`= seconds`

SET
This is a keyword for this command.

DEADTO
This is a keyword for this command.

seconds
This is the number of seconds that NetEx will wait until it disconnects a session because there was no response from the remote host.

SET DEFBI Command

The SET DEFBI command specifies the default maximum input buffer size for a connection. This default value is used if the user does not specify a maximum input buffer size in the CONNECT or OFFER request.

Command	Parameters
`SET DEFBI`	`= size`

SET
This is a keyword for this command.

DEFBI

This is a keyword for this command.

size
This is the default maximum input buffer size in words.

SET DEFBO Command

The SET DEFBO command specifies the default maximum output buffer size for a connection. This default value is used if the user does not specify a maximum output buffer size in the CONNECT or OFFER request.

Command	Parameters
`SET DEFBO`	`= size`

SET
This is a keyword for this command.

DEFBO
This is a keyword for this command.

size
This is the default maximum output buffer size in words.

SET DELAY0 Command

The SET DELAY0 command sets the initial one-way path delay when the configured path has DELAY=0 configured. This may help protocol 4 connections reach their optimum send speed faster. (See also SET SRATE0) (Protocol 4 is currently not supported.).

Command	Parameters
`SET DELAY0`	`= delay`

SET
This is a keyword for this command.

DELAY0
This is a keyword for this command.

delay
This is the initial one-way delay in milliseconds.

SET DRIVMAX

Specifies the maximum number of concurrent Driver Interface connections supported by NetEx. This count reflects only the number of DCONNECT's in effect. If a new DCONNECT is issued when DRIVMAX sessions are in progress, the DCONNECT request will be rejected with an error.

Command	Parameters
`SET DRIVMAX`	`= number`

SET
This is a keyword for this command.

DRIVMAX
This is a keyword for this command.

number
This is the maximum number of connections supported. The default is 0.

SET IDLETO Command

The SET IDLETO command specifies the amount of time that transport will wait before sending an idle message to verify the continued existence of a party at the other end of a logical connection. The transmission of any message resets the timer.

Command	Parameters
`SET IDLETO`	`= seconds`

SET
This is a keyword for this command.

IDLETO
This is a keyword for this command.

seconds
This is the number of seconds that NetEx will wait before sending an idle message to the remote host.

SET IPROUTE Command

The SET IPROUTE command associates a GNA address (from the NCT) to an IP address to reach the associated Netex on the network. Local GNA to IP mapping must also be specified.

Command	Parameters
`SET [ IPROUTE | IP ]`	`Gna Ipaddr`

SET
This is a keyword for this command.

IPROUTE or IP
This is a keyword for this command.

Gna
This is a local or remote GNA address (4 hex digits) from the NCT.

Ipaddr
This is a local or remote IP address (dotted decimal notation) for this GNA

EXAMPLE

```
SET IP 11C0 10.1.6.20
```

SET IPCLOG/IPCONS Commands

The SET IPCLOG command specifies the events that will be logged in print messages by the IPC program.

Command	Parameters
`SET [IPCLOG|IPCONS]`	`= level (0-8)`

SET

This is a keyword for this command.

IPCLOG or IPCONS

a keyword for this command.

level

- `0 - MSG_WARNING_CLASS`
- `1 - MSG_BANNER_CLASS`
- `2 - MSG_INFO_CLASS`
- `3 - MSG_TRACE_A_CLASS (Control requests and their completion status)`
- `4 - MSG_TRACE_B_CLASS(NTXn function requests and their completion status)`
- `5 - MSG_DEBUG_CLASS`
- `6 – 8 Reserved for future use.`

Messages of a CLASS greater than or equal to the current Filter Level are blocked. Setting level greater than 5 can create a large volume of print and affect performance. IPCLOG messages go to the IPC print file. IPCONS messages go to the console.

SET MAXBI Command

The SET MAXBI command specifies the maximum input buffer size that a user may specify on a CONNECT or OFFER call. This parameter sets a system wide maximum user buffer size. Its value and the size of the user buffer region determine possible fragmentation of the region and the maximum number of connections that can be supported.

Command	Parameters
`SET MAXBI`	`= size`

SET

This is a keyword for this command.

MAXBI

This is a keyword for this command.

size

This is the maximum input buffer size (in words) that users may specify on a CONNECT or OFFER call.

SET MAXBO Command

The SET MAXBO command specifies the maximum output buffer size that a user may specify on a CONNECT or OFFER call. This parameter sets a system wide maximum user buffer size. Its value and the size of the user buffer region determine possible fragmentation of the region and the maximum number of connections that can be supported.

Command	Parameters
`SET MAXBO`	`= size`

SET
This is a keyword for this command.

MAXBO
This is a keyword for this command.

size
This is the maximum output buffer size (in words) that users may specify on a CONNECT or OFFER call.

SET MAXOD Command

The SET MAXOD command specifies the number of octets of information a session user may specify in the NRBPROTL field in the NRB. This is used to send user protocol information. The maximum value is larger for transport, network, and driver level callers. It is increased by the maximum size of the previous level's protocol. This value is normally set to 256 but can be set to whatever value the application user needs (consistent with buffer usage).

Command	Parameters
`SET MAXOD`	`= bytes`

SET
This is a keyword for this command.

MAXOD
This is a keyword for this command.

bytes
This specifies the maximum length of user protocol data allowed. The default (and maximum) is 256 If LONGMSG is set the maximum is 144.

SET MSGLVL Command

The SET MSGLVL command controls the severity of messages printed on the operator's console. The operator messages are grouped from 0-15 (decimal). All messages with the specified level of severity or greater are displayed.

Command	Parameters
`SET MSGLVL`	`= number`

SET
This is a keyword for this command.

MSGLVL
This is a keyword for this command.

number
This is the minimum level of severity of messages to be displayed on the operator's console. Number may be any integer from 0 to 15:

MSGLVL	Description
12	Messages that require immediate action by the operator (e.g., network equipment failure, NetEx termination).
8	Messages that are of great interest to the operator and may require operator action. Examples: beginning and end of all sessions, intermittent failures of network equipment or communications media.
4	Messages regarding events that are of interest in closely monitored environments. Examples: memory shortage conditions, receipt of unsolicited messages from the network, display of statistics at the end of connections.
0	Messages that are intended for diagnostic or debugging purposes. These messages are generally only of interest when a system programmer is attempting to diagnose a NetEx problem.

SET NTXOPER Command

The SET NTXOPER command specifies if the remote operator service is to be enabled or disabled. It may also specify the number of remote operator sessions to allow.

Command	Parameters

`SET NTXOPER  | ON  |`
`             | OFF |`
`             | n |`
	`[ CLASS  | A | ]`
	`          | C |`
	`          | G |`

SET
This is a keyword for this command.

NTXOPER
This is a keyword for this command.

ON

This specifies that remote operator service is enabled (number allowed is increased by one).

OFF

This specifies that remote operator service is disabled (number allowed is decreased by one).

n

This represents the multiple remote operator sessions enabled.

CLASS

This is an optional keyword for this command. The class selected is the level of operator commands that will be accepted from the remote hosts. Class may also be selected using the SET ROPCLASS command. Valid classes include the following list.

A All commands (HALT, KILL, DRAIN included).

C Privileged class commands (SET, START, > (remote command)) and general class.

G General class commands (DISPLAY).

SET PIPELIM Command

The SET PIPELIM command sets the upper bound of total data that is allowed to be sent on a connection, regardless of the amount calculated by delay * rate.

Command	Parameters
`SET PIPELIM`	`= words`

SET

This is a keyword for this command.

PIPELIM

This is a keyword for this command.

words

This is the number of words allowed to be unacknowledged (in the pipe).

SET PREFPROT Command

The SET PREFPROT command selects the protocol type to use if both PROTOCOL=2 and PROTOCOL=4 are specified for a host. This was implemented to facilitate testing. (Protocol 4 is currently not supported.)

Command	Parameters
`SET PREFPROT`	`| 2 |` `| 4 |`

SET

This is a keyword for this command.

PREFPROT

This is a keyword for this command.

2 or 4

This specifies protocol type.

SET PRINT Command

The SET PRINT command controls the printing of messages into the NetEx log

Command	Parameters
SET PRINT	| ON | | OFF|

SET

This is a keyword for this command.

PRINT

This is a keyword for this command. The default is ON. Valid parameters are:

OFF Nothing will be written to the NetEx log,

ON Messages will be put into the NETEX log based on the current MSGLVL setting.

SET ROPCLASS Command

The SET ROPCLASS command specifies the class of operator commands that the remote operators will be allowed to issue.

Command	Parameters
SET ROPCLASS	| A | | C | | G |

SET

This is a keyword for this command.

ROPCLASS

This is a keyword for this command. Valid classes include:

A All commands (HALT, KILL, DRAIN included).

C Privileged class commands (SET, START, > (remote command)) and general class.

G General class commands (DISPLAY).

SET READTO Command

The SET READTO command specifies the number of seconds that NetEx transport will retain user data waiting for the receiver to issue a READ request. When this timer expires a disconnect will be sent to the remote process connected. The local process will be sent a disconnect message for READTO seconds, if there is not already one there. The transport connection will be cleared out and the Tref will become invalid for future user requests.

Command	Parameters
SET READTO	= seconds

SET

This is a keyword for this command.

READTO
This is a keyword for this command.

seconds
This is the number of seconds that NetEx transport will retain data waiting for a READ from the user.

SET SESMAX Command

The SET SESMAX command controls the number of session connections or OFFERs permitted at one time. If the current number of sessions is greater than the new value specified, the command will not affect sessions in progress but will deny any new requests until sessions are disconnected. If the current number of sessions is greater than the new value then there will be no immediate effect. SET SESMAX 0 will effectively drain session services (virtually identical to DRAIN NETEX).

Command	Parameters
`SET SESMAX`	`= number`

SET
This is a keyword for this command.

SESMAX
This is a keyword for this command.

number
This is the number of connections and OFFERs to allow concurrently.

SET SRATE0 Command

The SET SRATE0 command sets the initial transmit rate when the configured path has RATE=0 configured. This may help protocol 4 connections reach their optimum send speed faster. (See also SET DELAY0). (Protocol 4 is currently not supported.)

Command	Parameters
`SET SRATE0`	`= rate`

SET
This is a keyword for this command.

SRATE0
This is a keyword for this command.

rate
This is the initial rate in Kbytes/second

SET TRACE Command

The SET TRACE command specifies the trace options selected. The trace may be off, on memory or on tape, and on for either specified events or classes of events.

Command	Parameters
SET TRACE	ON [events] OFF [events] TAPE [reel#] DISK MEMORY FREEDISK FREETAPE

SET
This is a keyword for this command.

TRACE
This is a keyword for this command.

ON
This indicates that a trace is enabled. If events are specified, then only the event selection is changed, tracing remains ON or OFF as previously set.

OFF
This indicates that a trace is disabled. If events are specified, then only the event selection is changed, tracing remains ON or OFF as previously set.

events
These are the trace events to be saved. One or more trace events (see below) may be turned ON or OFF. If no events are selected, the currently active event set remains selected.

MEMORY
Memory will be used for the trace entries, in a circular fashion.

TAPE or DISK
Full buffers will be written to tape or disk, respectively (Filename: NETEX$TRACE).

Reel#
This specifies the tape reel number used for trace output. The tape is dynamically assigned.

FREETAPE
This causes the NETEX$TRACE tape to be freed. If tracing is ON, the output is then sent to memory.

FREEDISK
This causes the current NETEX$TRACE cycle to be freed. If tracing is ON, the output is then sent to memory.

Note: Setting trace OFF eliminates significant processing overhead for NetEx.

Table 6. Trace Events	
Trace Event	**Description**
ALL	All events listed in this table.
BA	Buffer Allocate
BF	Buffer Free
DA	Driver associated data
DI	Driver input (message proper only)
DO	Driver output(message proper only)
IC	I/O completion
IR	I/O request
MC	Module call
MR	Module return
OI	Operator command
OO	Operator output
QM	Queue manager
SP	Spawn
SH	Trash
SU	Suspend
UB	User buffer copied
UC	User request cleanup
UD	User request done
UR	User request in
UT	User request termination

SET WDOGINT Command

The SET WDOGINT command specifies the number of seconds that elapse between NetEx's checking for timed out conditions in the NRB requests. Thus, if a READ has a timeout value specified as 10 seconds, and the WDOGINT is also 10 seconds, the READ will actually timeout in the range 10 - 20 seconds.

Command	Parameters
`SET WDOGINT`	`= seconds`

SET

This is a keyword for this command.

WDOGINT

This is a keyword for this command.

seconds

This is the number of seconds that NetEx will use as a base unit for timing.

Appendix A. NRBSTAT Error Codes

Whenever a NetEx request is issued, the results of the request are returned in one or both of two fields, NRBSTAT and NRBIND. These are located at the beginning of the NRB to make their subsequent examination by high level language programs a simpler matter.

NRBSTAT is designed to indicate if an operation is in progress, and whether it completed successfully or not. NRBIND is designed to indicate the type of information that arrived as the result of a read-type command (OFFER or READ).

When the operation is accepted by the NetEx user interface, the value of NRBSTAT is set to the local value of –1. Thus, the sign of this word is an "operation in progress" flag for all implementations.

If an operation completed successfully, NRBSTAT will be returned as all zeroes. If a read-type command was issued, then an "indication" will be set in NRBIND.

If the operation did not complete successfully, then NRBSTAT will contain a standard error code. NRBSTAT is represented as a decimal number that is potentially as large as 2^{15-1} (32,767). The 2^{16} bit is not used so that it may remain an "in progress" flag on the 16 bit machines. The thousands digit denotes the origin of the error; the low order three digits specifically identify the error type. The codes for error origin are as follows:

0xxx NetEx general; errors detected by the user interface that prohibit proper processing of the command.

09xx Reserved for implementation dependent errors in the user interface.

1xxx Driver level errors.

2xxx Transport level errors.

3xxx Session level errors.

4xxx Network level errors.

5xxx-8xxx Reserved for future NetEx functions.

90xx Reserved for errors returned by User Exits on the local host.

91xx Reserved for errors returned by User Exits on a remote host during the connection process.

9200-32767 Reserved

0xxx and 90xx errors can be returned to any user program that accesses NetEx services. Normally, an application that accesses services at the session level will only receive those errors (3xxx) related to session services. However, the principle within NetEx should be that if a level elects to abort the user's request based on an error returned by a lower level of software, then the error code should be "rippled up" to the user rather than summarized at the higher level. For example, driver might report a "power off" or "not operational" status to transport in the event of an adapter failure. If transport decided that this error type should cause loss of communications, then the 1xxx error would be returned to the user along with a Disconnect Indication in NRBIND when the next user read command was issued.

Following that, the second digit places the errors in categories.

x0xx NetEx general or inconsistent NRB format.

x1xx Specification errors in parameters passed to a particular protocol level.

x2xx Hardware errors.

x3xx Requests out of sequence and read timeouts.

x4xx NetEx initiated disconnect errors.

x5xx Errors during connection.

The error codes at each level have been made as common as possible. Thus, a 2103 error in transport would have substantially the same meaning as a 3103 error in session, and a 1361 error would not be defined at (for example) the Driver level if a 3361 error meant something entirely different at the Session level.

Finally, some errors will cause the loss of the connection or will result in a connection not being established in the first place. Any status code that implies that the connection is no longer useful will have a 6 (Disconnect Indication) returned in NRBSTAT. Any attempts to issue further requests to that connection will have a xl00 (no N-ref) error returned to it. All errors that result in loss of the connection and a Disconnect Indication in NRBIND are indicated by an asterisk (*) following the error code number.

Note: A 0000 in field NRBSTAT means successful completion of NetEx request. A -l means that request is still in progress.

The following subsections describe the errors in numerical order starting with general NetEx errors, followed by driver, transport, and session level errors.

General Errors

The following errors are general NetEx or user interface errors:

1 A read-type operation completed normally within NetEx, but the buffer provided by the user was not large enough to hold the data. NRBLEN and NRBUBIT reflect the amount of data the other party intended to send; however, the amount of data moved to the user's program was only the amount of addressable units specified in NRBBUFL. NRBIND specifies the type of data sent to the user. Requests affected: SREAD or SOFFER. The status of the connection is not affected.

2 NRBBUFL and NRBBUFA do not specify a block of storage that fits entirely within the user's addressable memory. The operation is suppressed. All requests are affected. The status of the connection is unaffected.

3 On a write type operation, the unused bit count (NRBUBIT) specifies a larger number of bits than are in the machine's word (addressable unit size). The operation is suppressed; the status of the connection is not affected.

4 The request code (NRBREQ) is not valid. The operation is ignored, and the status of the connection specified by NRBNREF is not affected.

5 The buffer size specified (in NRBBUFL for read and NRBLEN for a write) exceeds an implementation defined NetEx maximum. The operation is suppressed. The status of the connection is not affected.

11 A read-type operation completed normally within NetEx, but the Odata buffer provided by the user was not large enough to hold the data. NRBPROTL reflects the amount of data the other party intended to send; however, the amount of data moved to the user's program was only the amount of addressable units originally specified in NRBPROTL, NRBIND specifies the type of data sent to the user. Requests affected: xOFFER, xREAD. The status of the connection is not affected.

12 NRBPROTL and NRBPROTA does not specify a block of storage that fits entirely within the user's addressable memory. The operation is suppressed. All requests are affected. The status of the connection is unaffected.

15 The buffer size specified (in NRBPROTL) exceeds an implementation defined NetEx maximum. The limit in NetEx is 144 bytes with long message enabled

21 A read-type operation completed normally within NetEx, but both the Odata and Pdata buffers were too small to hold the incoming data. NRBLEN/NRBUBIT and NRBPROTL reflect the amount of data the other party intended to send; however, the amount of data moved to the user's program was only the amount of addressable units originally specified in NRBLEN and NRBPROTL. NRBIND specifies the type of data sent to the user. The requests affected are xOFFER and xREAD. The status of the connection is not affected.

100* The user interface detected that the N-ref for that level of service is not currently in use by the application program. Probable cause is a bad CONNECT, OFFER, or ASSIGN or failure to handle an incoming DISCONNECT.

310 The user has attempted to re-use an NRB before a previous request issued with that NRB has completed. The request will be rejected. When the original request issued with that NRB completes, then the NRB will be once more updated with the status of that request.

500* NetEx is not currently running on the local computer. Intervention by the local computer operator will be required to start NetEx. This code is issued by the NetEx user interface when it is determined that NetEx is unavailable.

503* An OFFER, CONNECT, or ASSIGN request has resulted in the number of connections outstanding for the caller exceeding an implementation defined maximum. The new connection request is rejected.

504* The user program is not authorized to use the user interface facilities needed to communicate with NetEx. No use of NetEx is possible until the user gains the appropriate authorization.

505* NetEx is currently being drained by the computer operator in preparation for a NetEx shutdown. No new OFFER, CONNECT, or ASSIGN requests will be accepted. The request is rejected. The status of already existing connections is not affected.

511* A CONNECT or ASSIGN request would exceed the total number of driver service level connections to NetEx. The new connection request is rejected.

512* The NetEx program is aborting execution due either to internal NetEx software problems or cancellation by the computer operator. No further traffic with NetEx will be possible. This error will be issued to complete a request that was issued when NetEx was running normally.

Special NRBSTAT Errors for Unisys NetEx

901	NetEx-user protocol error.
902	Maximum concurrent outstanding reads/writes exceeded.
906*	Unable to issue TRMRG$ for user.
909	No buffer available for session control table
911*	NREF in use for Network/Transport call for Connect/Offer.
912	Level error. For example, Network level call after Transport level connect.
913	Maximum user requests exceeded.
914	Maximum total system requests exceeded.
915	Interface level error. Old NXIFTN calling new NTX$COMN bank.
916	Attempt to change common D-banks. The user must use the same NetEx common D-bank during an execution.
917	Different utility D-bank on successive NetEx calls.
920	Error during data transfer between NetEx and user. (It is most likely, the user LCORE'd away the buffer address space that NetEx is writing into.)
921	Stop-bit in A-format data caused truncation. (Remove bad data or use a different datamode).
922	Data translation may not have been done. The hardware/software translation for the specified character set is not available on this host. Check NRBDMODE for current (outgoing) and desired (incoming) character sets.
925	The path selection routine selected a path number that was larger than the number of paths in the pam entry. The error condition is returned to the user.

Driver Errors

1005 The buffer size specified (in NRBBUFL for read and NRBLEN for a write) exceeds an implementation defined NetEx maximum. The operation is suppressed. The status of the connection is not affected.

1100* The D-ref specified by NRBNREF is not in use or is not owned by this application program. The request is rejected. The status of other connections owned by this application remain unchanged.

1101 The DATAMODE field of a NetEx format network message is not valid for this particular host. DWRITE for NetEx format messages are affected. The driver assignment remains in effect.

1102 The specified value of the Associated Data bit in the hardware message area does not match the presence or absence of associated data as specified in NRBLEN. DWRITE is affected. The driver assignment remains in effect. Both NetEx format and arbitrary format network messages are affected.

1103 The specified length of the message proper does not fit within the HYPERchannel-imposed limits of 8 to 64 bytes inclusive. Only DWRITEs may obtain this response. The driver assignment remains in effect. Both NetEx format and arbitrary format network messages are affected.

1200 "Power off," "not operational," or a similar indication of local adapter unavailability was discovered when physical I/O was issued. The status of the assignment is not affected, but it is unlikely that driver communications can continue without operator intervention.

1201 The network adapter has reported an error in processing the DREAD or DWRITE request. The adapter model dependent detailed status may be obtained by issuing a DSTATUS function.

1202 I/O halted by DFREE request.

1300 A DREAD request timed out before any data was received on the network. The time value used for the timeout was in NRBTIME. No data was received. The status of the driver connection is not affected.

1304 The number of DWRITE requests outstanding against a single connection exceeds an implementation defined maximum (usually one.) The DWRITE request is rejected. The status of the connection and the previous DWRITE requests remains unchanged.

1305 The number of DREAD requests outstanding against a single connection exceeds an implementation defined maximum (usually one.) The DREAD request is rejected. The status of the connection and the previous DREAD request remains unchanged.

1310 The device service was forced to discard the Associated Data segment of a message because no DREAD was issued within a sufficient time of the arrival of the network message. The message proper is returned to the user. Also, a DWRITE will receive this error if an intrahost DWRITE cannot be matched with an outstanding DREAD by another driver user; in that case, the message proper will be queued and the associated data discarded.

1311 Message Propers have been lost due to excess demand for the driver service's resources. One or more messages that arrived before the current message were discarded by the driver service. This will be provided as a DREAD error or an intrahost DWRITE.

1312 A request for a privileged service such as diagnostic mode has been issued to the driver. The request is rejected as the user does not have sufficient implementation dependent privileges. This error is applicable to both DREAD requests and intrahost DWRITE's.

1313 The total number of system-wide driver request has exceeded an installation defined maximum. The request will be rejected until the total number of outstanding driver requests drops below this maximum. If the problem persists, then the memory requirements of the driver should be increased to accommodate the greater number of concurrent requests.

1501* A specific D-ref requested by the TASSION or DASSIGN is already in use. If a nonspecific request was made, all driver paths are in use.

1503* The number of user driver attaches permitted by NetEx has been exceeded. Driver service cannot be offered at this time. The DASSIGN is rejected.

1504* Driver service is not directly available to applications programs. This service can only be made available by the installation systems programmer.

1505* NetEx is currently being "drained" by the computer operator. No new Driver service (DCONNECT) requests are being accepted.

1506* The specific D-ref (adapter address) requested by a DCONNECT does not exist on this local host.

1507* The specific D-ref (adapter address) exists on the local host, but the NetEx operator has drained or halted that adapter so no new requests for Driver service (DCONNECT requests) can be accepted on that adapter.

Transport Errors

2005 During a WRITE operation, the length of the buffer as specified by NRBLEN exceeds the maximum buffer size found in NRBBLKO. The WRITE operation is rejected. The connection remains outstanding. This error may also occur when the NetEx host names you are transferring data between are the same NetEx host name and the blocksize is greater than the segment size. Using NTXLCL as the host name will allow the transfer to successfully complete.

2007 ODATA size exceeds maximum.

2100* The T-ref specified by NRBNREF is not in use or is not owned by this application program. The request is rejected. The status of other connections owned by this application remain unchanged.

2101 The DATAMODE field in the NRB is not valid for the local host. The write operation (SWRITE, TWRITE, SCONNECT, TCONNECT) is suppressed. The connection (if previously established) remains in effect.

2300 The timeout value associated with a TREAD request resulted in a request tuning out before any data or other indication was received from the corresponding application.

2301 TCONNECT, TOFFER, or TCONFIRM has been issued for a connection that is already fully established. The request is rejected. The status of .the connection remains unchanged. Some implementations may return a 2301 code for any “out of sequence” series of requests to Transport.

2302 A Connect Indication was received by a preceding offer, and a request other than TCONFIRM or TDISCONNECT was issued. The request is rejected. NetEx will continue to wait for the confirm or disconnect request.

2303 A TCONNECT request was previously issued, and then a request other than a TREAD to read the Confirm or Disconnect Indication was issued. The request is rejected. NetEx will continue to wait for the TREAD request.

2304 The number of TWRITE requests outstanding against a single connection exceeds an implementation defined maximum (usually one.) The TWRITE request is rejected. The status of the connection and the previous TWRITE requests remains unchanged.

2305 The number of TREAD requests outstanding against a single connection exceeds an implementation defined maximum (usually one). The TREAD request is rejected. The status of the connection and the previous TREAD requests remains unchanged.

2306 A TWRITE request has been issued to a Transport connection that is in the process of servicing a remote caller or NetEx initiated Disconnect. A Disconnect Indication is pending from NetEx.

2307* A TREAD request has been issued to a Transport connection that is in the process of servicing a remote caller or NetEx initiated Disconnect. A Disconnect Indication is pending from NetEx.

2308 A write type request (TWRITE or another TCLOSE) has been issued against a connection that has accepted a previous TCLOSE.

2400* No response has been received from the remote NetEx for a period of elapsed time (DEADTO) specified by the installation systems programmer. The connection is terminated. A Disconnect Indication will be found in NRBIND.

2401* The calling application has failed to issue a read request to NetEx for a period of elapsed time (READTO) specified by the installation systems programmer. The connection is terminated for both parties. The local NetEx will retain this 2401 error for another READTO seconds, at which point a subsequent read will result in a 2100 (no T-ref) error.

2402* The remote application has failed to issue a TREAD request for a period of elapsed time (READTO) specified by the installation systems programmer on the remote host. The connection is terminated. A Disconnect Indication will be found in NRBIND.

2403* The remote application exited without issuing an explicit Disconnect back to the local application. The connection is terminated. A Disconnect Indication will be found in NRBIND.

2500* A connect message was repeatedly sent to the remote host in response to a previous TCONNECT request, but no response was received for a period of elapsed time (CONTO) specified by the installation systems programmer.

2501* A specific N-ref requested by the TCONNECT is already in use.

2503* The number of user Transport connections permitted by NetEx has been exceeded. Transport service cannot be offered at this time. The TCONNECT is rejected.

2504* Transport service is not directly available to applications programs. This service can only be made available by the installation systems programmer.

2505* NetEx is currently being “drained” by the computer operator. No new requests for Transport services (TCONNECT requests) are being accepted.

2506* The Physical Address Map passed to Transport for a connection is not valid. If returned from an SCONNECT request, it is due to an incorrectly generated Network Configuration list.

2509* The specified value of NRBBLKO exceeds an installation or implementation defined maximum. The connection request is rejected.

2510* The specified value of NRBBLKI exceeds an installation or implementation defined maximum. The connection request is rejected.

2511* The specified class of service is not implemented.

Session Errors

3005 During a WRITE operation, the length of the buffer as specified by NRBLEN exceeds the maximum buffer size found in NRBBLKO. The WRITE operation is rejected. The connection remains outstanding.

3006 The length of PDATA sent on a CONNECT, CONFIRM, or DISCONNECT is greater than the maximum allowed. The request is rejected.

3100* The S-ref specified by NRBNREF is not in use or is not owned by this applications program. The request is rejected. The status of other connections owned by this application remains unchanged.

3101 On an SWRITE request for intra-host communications, a DATAMODE was specified that is not supported for internal communications.

3103 The quantity of Odata provided exceeds an implementation defined maximum. The request is rejected.

3300 An SREAD or SOFFER request timed out before a response was received on the network. If the timed request is an SREAD, the status of the connection was not affected. If a SOFFER timed out, then the connection will not have taken place.

3301 SCONNECT or SCONFIRM has been issued for a connection that is already fully established. The request is rejected. The status of the connections remains unchanged.

3302 A Connect Indication was received by a preceding SOFFER, and a request other than SCONFIRM or SDISCONNECT was issued. The request is rejected. NetEx will continue to wait for the confirm or disconnect request.

3303 Wait for SREAD or SDISCONNECT.

3304 The number of SWRITE requests outstanding against a single connection exceeds an implementation defined maximum (usually one.) The SWRITE request is rejected. The status of the connection and the previous SWRITE requests remains unchanged.

3305 The number of SREAD requests outstanding against a single connection exceeds an implementation defined maximum (usually one.) The SREAD request is rejected. The status of the connection and the previous SREAD requests remains unchanged.

3306 An SWRITE request has been issued to a session connection that is in the process of servicing a remote caller or NetEx initiated Disconnect. A Disconnect Indication is pending from NetEx.

3307 An SREAD request has been issued to a session connection that is in the process of servicing a remote caller or NetEx initiated Disconnect. A Disconnect Indication is pending from NetEx.

3308 A write type request (SWRITE or another SCLOSE) has been issued against a connection that has accepted a previous SCLOSE.

3401* The calling application has failed to issue a read request to NetEx for a period of elapsed time (READTO) specified by the installation systems programmer. The connection is terminated for both parties. The local NetEx will retain this 3401 error for another READTO seconds, at which point a subsequent read will result in a 3100 (no S-ref) error.

3402* The remote application has failed to issue an SREAD request for a period of elapsed time (READTO) specified by the installation systems programmer on the remote host. The connection is terminated. A Disconnect Indication will be found in NRBIND.

3403* The remote application exited without issuing an explicit Disconnect back to the local application. The connection is terminated. A Disconnect Indication will be found in NRBIND.

3422* A HALT SREF operator command was issued against this session.

3500* A connect message was repeatedly sent to the remote host in response to a previous TCONNECT request, but no response was received for a period of elapsed time (CONTO) specified by the installation systems programmer. Probable cause is the absence of the NetEx software on the remote host. The SCONNECT terminates with a Disconnect Indication in NRBIND.

3501* The PNAME specified is not OFFERed on the HOST specified during the SCONNECT. The SCONNECT terminates with a Disconnect Indication in NRBIND.

3502* The PNAME specified is not OFFERed on the HOST specified during the SCONNECT. However, a session that was previously established by OFFERing the requested PNAME is now in progress on the remote machine. If the remote application elects to re-OFFER the connection in the future, the service might be available at that time. (In other words, the remote application is “busy.”)

3503* The number of user session connections permitted by NetEx has been exceeded. Session service cannot be offered at this time. The SCONNECT or SOFFER is rejected.

3504* Session service is not directly available to applications programs. This service can only be made available by the installation systems programmer.

3505* NetEx is currently being “drained” by the computer operator. No new requests for Session services (SCONNECT and SOFFER) are being accepted.

3506* The HOST specified in an SCONNECT request does not exist anywhere on the network generated by the installation systems programmer. The SCONNECT terminates with a Disconnect Indication in NRBIND.

3507* The HOST specified exists on the installation generated network configuration, but the local computer operator has specified that no session level connections take place with that particular host. The SCONNECT terminates with a Disconnect Indication in NRBIND.

3508* The HOST specified exists on the installation generated network configuration, but no communications path exists between the local host and the specified remote host. The SCONNECT terminates with a Disconnect Indication in NRBIND.

3509* The specified value of NRBBLKO exceeds an installation or implementation defined maximum. The connection request is rejected.

3510* The specified value of NRBBLKI exceeds an installation or implementation defined maximum. The connection request is rejected.

3511* The class of service requested is not currently implemented.

3516 The HOST specified in the SCONNECT protocol received by the session manager is not the name of the local HOST.

3522 Netex is in the process of draining, and this offer was still outstanding.

3530 Attempt by a user to use an internal reserved name. Currently, names beginning with NTX (for example, NTXOPER) are reserved.

Network Service Errors

4005 The buffer size specified in NRBBUFL for read or NRBLEN for a write exceeds an implementation defined NetEx maximum. The operation is suppressed. The status of the connection is not affected.

4015 The buffer size specified in NRBPROTL exceeds an implementation defined NetEx maximum.

4100* The N-ref specified by NRBNREF is not in use or is not owned by this application program. The request is rejected. The status of other connections owned by this application remain unchanged.

4101 In a Network connection that is intra-host (causing no network adapter traffic) a DATAMODE was requested on the NWRITE that is not supported for intra-host communications. The block will be sent to the destination process using bit stream (DATAMODE 0) transmission.

4104 Checksum on an incoming driver level message is not correct. The message and data received will be returned to the NREAD caller along with the error code but the data should, of course, be considered suspect. The status of the driver assignment is not affected.

4105 The length of the Pdata was less than, or substantially different from, the specified length in the message proper. This comparison is performed after adjustment for incoming A/D modes. Sufficient slop in this comparison will be done to accommodate those machines that must send information in multiples of the word size.

4300 The timeout value associated with an NREAD request resulted in a request timing out before any data or other indication was received from the corresponding application.

4301 NCONNECT or NOFFER has been issued for a connection that is already fully established. The request is rejected. The status or the connection remains unchanged. Some implementations may return a 4301 code for any “out of sequence” series of requests to network service.

4304 The number of NWRITE requests outstanding against a single connection exceeds an implementation defined maximum (usually one.) The NWRITE request is rejected. The status of the connection and the previous NWRITE requests remains unchanged.

4305 The number of NREAD requests outstanding against a single connection exceeds an implementation defined maximum (usually one). The NREAD request is rejected. The status of the connection and the previous NREAD requests remains unchanged.

4306 An NWRITE request has been issued to a Transport connection that is in the process of servicing a NetEx initiated Disconnect. A Disconnect Indication is pending from NetEx.

4307 An NREAD request has been issued to a Transport connection that is in the process of servicing a NetEx initiated Disconnect. A Disconnect Indication is pending from NetEx.

4403 When processing an NWRITE request, Network service found that a Network Virtual Circuit between the two Network applications no longer exists. The Network connection is terminated.

4501* A specific N-ref requested by the NCONNECT or NOFFER is already in use.

4503* The number of user Network connections permitted by NetEx has been exceeded. Network service cannot be offered at this time. The NCONNECT or NOFFER is rejected.

4504 Network service is not directly available to applications programs. This service can only be made available by the installation systems programmer.

4505* NetEx is currently being “drained” by the computer operator. No new requests for Network services (NCONNECT or NOFFER requests) are being accepted.

4506 The Physical Address Map passed to Network for a connection is not valid. If returned from an SCONNECT request, it is due to an incorrectly generated Network configuration list.

4509 The specified value of NRBBLKO exceeds an installation or implementation defined maximum. The connection request is rejected.

4510* The specified value of NRBBLKI exceeds an installation or implementation defined maximum. The connection request is rejected.

4511* The specified class of service is not implemented.

Appendix B. ConfMang Messages

First Pass Configuration Messages

The following is a list of warning messages that can be displayed by the configuration manager parser during pass one. These errors are not fatal and result in printing an error message and continuing the parse. All messages except CONF00I, CONF005, CONF006 and CONF029 set the NCT validation flag (ValidNCT) to FALSE. This flag is checked before accessing the NCT. The message, explanation, and action are discussed below. The actual value of the identifier will replace an identifier enclosed in single quotes (for example: "HOSTname").

CONF001I Parsing initiated
Explanation: This is only an informational message.
System Action: Display message and continue processing.

CONF002E Invalid configuration statement
Explanation: An unrecognizable statement type was encountered. The statement that is invalid is displayed followed by the error message.
System Action: Ignore card and get next one.

CONF003E 'name' statement: 'label' encountered after LINK statement: 'label'
Explanation: A statement was encountered after a LINK statement. The order of these statements is wrong.
System Action: Ignore statement, free up memory allocated to 'name' get next card.

CONF004E TRUNK not previously defined
Explanation: Tx = label was encountered in an ADAPTER or LINK statement and the trunk referred to by the label has not been defined.
System Action: Ignore operand and scan for next one.

CONF005I Syntax check halted; the following not checked
Explanation: An error was encountered that resulted in some code to be overlooked. The statements following this message are NOT checked for errors.
System Action: Skip code until a key word is found. This is only a warning.

CONF006I Syntax check resumed; the previous stint checked
Explanation: A key word was encountered in the previous statement. The statement previous to this message is checked for errors.
System Action: Inform user of code not checked. This is only a warning.

CONF007E Statement missing label: 'TRUNK | HOST | PORT'
Explanation: The label is missing from the TRUNK, HOST or PORT statement.
System Action: Ignore card and get next one.

CONF008E Invalid operand: 'operand'
Explanation: An unrecognizable operand was encountered.
System Action: Ignore operand and get next one.

CONF009E Invalid network type: 'operand'
Explanation: An unrecognizable network type was encountered in the LOCALNET statement,
System Action: Ignore statement and get next card.

CONF010E TYPE missing in LOCALNET statement: 'network name'
Explanation: The type operand was missing from the LOCALNET statement. It is required for all networks.
System Action: Free up memory allocated to network and get next card.

CONF011E NETADDR missing in ADAPTER/LINK statement: 'label'
Explanation: The NETADDR operand is missing from the ADAPTER or LINK statement. It is required for all adapters/links.
System Action: Ignore statement, free up memory allocated to adapter or link and get next card.

CONF012E MODEL missing in ADAPTER/LINK statement: 'label'
Explanation: The MODEL operand is missing from the ADAPTER or LINK statement. It is required for all adapters/links.
System Action: Ignore statement, free up memory allocated to adapter or link and get next card.

CONF013E TRUNK missing from ADAPTER/LINK statement: 'label'
Explanation: A trunk operand is missing from the ADAPTER or LINK statement. At least one must be specified (and not more than 4). It is required for all adapters/links.
System Action: Ignore statement, free up memory allocated to adapter or link and get next card.

CONF014E CHANADDR missing in ADAPTER statement: 'label'
Explanation: The CHANADDR operand is missing from the ADAPTER statement
System Action: Ignore statement, free up memory allocated to adapter and get next card.

CONF015E RATE invalid in PORT statement: 'rate'
Explanation: An invalid rate was specified in a PORT statement.
System Action: Ignore statement, free up memory allocated and get next card.

CONF016E SMGDREF: 'hexvalue' must have rightmost 2 bits = PORT: 'value'
Explanation: A DREF was encountered that has the value 'hexvalue'. The low-order 2 bits must equal the port number 'value'.
System Action: Ignore statement, free up memory allocated and get next card.

CONF017E DEST missing in PORT statement: 'label'
Explanation: The DEST operand is missing from the PORT statement. It is required for all adapters.
System Action: Ignore statement, free up memory allocated to port and get next card.

CONF018E RATE missing in PORT statement: 'label'
Explanation: The RATE operand is missing from the PORT statement. It is required for the A142 Adapters.
System Action: Ignore statement, free up memory allocated to port and get next card.

CONF019E Invalid adapter model: 'model'
Explanation: The MODEL operand specified an invalid adapter model number in the ADAPTER or LINK statement.
System Action: Ignore statement, free up memory allocated to adapter or link and get next card.

CONF020E NUMADDRS encountered without CHANADDR: 'label'
Explanation: The NUMADDRS operand was encountered but no CHANADDR was previously defined.
System Action: Ignore statement, free up memory allocated to adapter and get next card.

CONF021E Invalid PORT number 'number'
Explanation: The PORT operand is out of the range (0...3) in an ADAPTER or LINK statement.
System Action: Ignore statement, free up memory allocated to adapter or link and get next card.

CONF022E Unexpected End of File Encountered
Explanation: The end of the configuration file was encountered before the completion of the parse.
System Action: Exit from module.

CONF023E TRUNK not meaningful In HYPERbus environment: 'trunk name'
Explanation: The TRUNK statement is not to be used in describing a HYPERbus network.
System Action: Ignore statement, free up memory allocated to adapter or link and get next card.

CONF025E Invalid hexadecimal character: 'char' in string: 'string'
Explanation: An invalid hexadecimal character was encountered.
System Action: Ignore statement, free up memory allocated to adapter or link and get next card.

CONF026E Invalid INTEGER: 'number'
Explanation: An invalid integer was encountered in parse.
System Action: Ignore statement, free up memory allocated to PORT and get next card.

CONF027E Invalid PROTOCOL level: 'number' on HOST: 'HOSTname'
Explanation: A protocol level was encountered that is not currently available.
System Action: Ignore statement, free up memory allocated to HOST and get next card.

CONF028E 'parameter name' parameter previously defined as 'name'
Explanation: A parameter was previously encountered and defined to be 'name'.
System Action: Ignore statement, free up memory allocated and get next card.

CONF029I Variable truncated: 'variable'
Explanation: A variable was encountered that is longer than the specified length for that variable.
System Action: Truncate to specified length and continue. This is only a warning.

CONF030E 'name1' statement: 'label' encountered before 'name2' statement: 'label'
Explanation: The statement specified by 'name1' was encountered before the statement specified by 'name2'. The statement label is specified by 'label'.
System Action: Ignore statement, free up memory allocated to HOST and get next card.

CONF03IE Label: 'label' and ADAPT: 'adapt' may not both be specified
Explanation: Both a label and an ADAPT parameter may not be specified for the adapter/link.
System Action: Ignore statement, free up memory allocated and get next card.

CONF033E token too long: 'string'
Explanation: A token was encountered that was longer than the maximum token length.
System Action: Ignore statement, free up memory allocated and get next card.

CONF034E Invalid Return Code
Explanation: An invalid return code was returned from the translator phase of SCANNER.
System Action: Ignore statement, free up memory allocated and get next card.

CONF070E Invalid SMGn-ref: 'hexvalue', should always = FFFF
Explanation: An n-ref was encountered with the value 'hexvalue'. SMGn-refs should always be 'FFFF'.
System Action: Ignore statement, free up memory allocated and get next card.

CONF071E SMGDREF: 'hexvalue' must have rightmost 2 bits = PORT: 'value'
Explanation: A DREF was encountered with the value 'hexvalue'. The low-order 2 bits must equal the port number 'value'.
System Action: Ignore statement, free up memory allocated and get next card.

CONF073E Invalid 710-mode address: 'address'
Explanation: The 710-mode address is not in the range '00' – 'FF'.
System Action: Ignore statement, free up memory allocated and get next card.

CONF074E Invalid trunk (0,1) defined on 715 link 'trunk', NETADDR= 'ad'
Explanation: 715 links may only have trunks 2 and/or 3 attached.
System Action: Ignore statement, free up memory allocated and get next card.

CONF080E BUS not previously defined: ‘busname’
Explanation: A BUS = parameter has been encountered, but the name does not match any previously defined HB statements.
System Action: Ignore statement, free up memory allocated and get next card.

CONF081E BUS not meaningful with “A” type adapter: ‘busname’
Explanation: “A” type adapters may not be attached to a BUS.
System Action: Ignore statement, free up memory allocated and get next card.

CONF082E BUS missing in ADAPTER/LINK statement: ‘netaddr’
Explanation: “B” type adapters must be attached to a BUS.
System Action: Ignore statement, free up memory allocated and get next card.

CONF083E Invalid SEGSIZE value: ‘value’
Explanation: SEGSIZE as specified on a PORT statement is > 65535.
System Action: Ignore statement, free up memory allocated and get next card.

CONF087F Line length exceeds 64 characters
Explanation: The line that follows is longer than 64 characters.
System Action: Edit the line to 64 characters.

Second Pass Configuration Messages

A formal description of the pass two configuration errors follow. The actual value of the identifier will replace an identifier enclosed in single quotes (for example: 'HOSTname'). All errors except CONF044, CONF046, CONF048, and CONF054, and CONF057 set the NCT validation flag (ValidNCT) to FALSE. This flag is always checked before accessing the NCT.

CONF03SE Multiple adapters must have same 'operand' value
Explanation: A multi-port adapter was encountered with duplicate NETADDRs but the one of the other operands don't match in both statements.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF036E ADAPT operand: 'operand' must equal adapter label: 'label'
Explanation: A multi-port adapter/link was encountered with duplicate NETADDRs but the ADAPT operand doesn't match the previous label.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF037E Multi-port adapters must not have same PORT
Explanation: A multi-port adapter was encountered with duplicate NETADDRs but the PORT operands are not different.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF03SE Duplicate labels in configuration: 'label'
Explanation: An entity was encountered with the same label as a previously defined entity.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF039E No adapter specified for HOST: 'HOSTname'
Explanation: No adapter was specified for this host. At least one is required for each host.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF040E No SMGR specified for HOST: 'HOSTname':
Explanation: No SMGR was specified for this host. At least one is required for each host.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF043E HOST: 'HOSTname' has GROUP name equal to real HOST name: 'name'
Explanation: A GROUP name was encountered that has the same name as a real HOST name.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF044I No PORT found with label: 'label'
Explanation: A DEST label was encountered and the PORT referred to by the label is not defined.
System Action: Display error message and continue processing. This is only a warning message.

CONF045E Invalid PORT linkage: 'link model', 'link model'
Explanation: A PORT was encountered that is not a valid linkage to the PORT described in the DEST label. PORTs must connect LINKs of the same type except for x4xx links.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF046I No path from HOST 'HOSTname' to HOST 'HOSTname'
Explanation: There is no path connecting HOST A with HOST B. Communication between these hosts cannot take place.
System Action: Display error message and continue processing. This is only a warning message.

CONF047E Missing a multi-port ADAPTER with the label: 'label'
Explanation: There is no adapter with the specified label; there must be for all x4xx adapter/links.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF048I The erroneous values are ‘value’ and ‘value’
Explanation: An error was encountered in which these two values are incorrect.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF049E Multi-port adapters can’t have 2 labels ‘label’, ‘label’
Explanation: An error was encountered in which these two values are incorrect.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF050E Multi-task adapters require unique SMGDREFs
Explanation: On a multi-process host, two adapters were encountered with the same NETADDR and the same SMGDREF. The SMGDREF values must be different. See ADAPTER statement section for details on multi-process hosts.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF051E Multi-task adapters require same PORT numbers
Explanation: On a multi-process host, two adapters were encountered with the same NETADDR and different PORT numbers. The PORT values must be the same. See ADAPTER statement section for details on multi-process hosts.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF053E Duplicate network address: ‘address’ in LOCALNET: ‘label’
Explanation: An adapter other than model PB40x/NB40x was encountered which specifies the same NETADDR as a previously defined adapter on the same network.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF054I The above NETADDR = an implicitly defined A722
Explanation: The NETADDR specified in the above error message is the same as the A722 unit of an S720 SLS.
System Action: Display error message and continue processing. This is only a warning message to explain the previous error further.

CONF055E HOST: ‘name’ and HOST: ‘name’ in GROUP ‘name’ need same protocols
Explanation: All hosts in the same group must have specified the same session manager DREF.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

CONF057I Parsing completed ValidNCT= ‘flag’, Errors= ‘number’, Warnings= ‘number’
Explanation: This is only an informational message. The state of the ValidNCT flag is displayed along with the number of error and warning messages encountered.
System Action: Display message and continue processing.

CONF072E Multi-task adapters require unique CHANADDRs
Explanation: On a multi-process host, two adapters were encountered with the same NETADDR and the same CHANADDR. The CHANADDR values must be different. See ADAPTER statement section for details on multi-process hosts.
System Action: Display error message, set ValidNCT flag to FALSE and continue processing.

MAKEPAM Processing Messages

The errors that could be encountered in CONFgetPAM (while processing the MAKEPAM command) are as follows. The actual value of the identifier will replace an identifier enclosed in single quotes (for example: 'HOSTname').

CONF058E HOST not defined in global configuration: 'HOSTname'
Explanation: The HOST specified was not found in the NCT.
System Action: Generate error message. No PAM is returned to caller.

CONF059E NCT is invalid, PAM cannot be built
Explanation: The parser has detected one or more errors. Therefore, the PAM cannot be built.
System Action: Generate error message. No PAM is returned to caller.

CONF060E No path from HOST: 'HOSTname' to HOST: 'HOSTname'
Explanation: No path exists from the specified local host to the specified remote host.
System Action: Generate error message. .

CONF061E NCT is invalid, HOST configuration cannot be built
Explanation: The parser has detected one or more errors; therefore, this entry cannot be built.
System Action: Generate error message. No entry is returned to caller.

CONF099I Path exists from HOST: 'hostname' to HOST: 'hostname'
Explanation: There is a route from the first 'hostname' to the second. Informational message only.
System Action: Generate message and continue processing.

Appendix C. NetEx Messages

NetEx messages are displayed on the system console or on the NetEx operator's terminal. The NetEx messages are described and suggested action to be taken is detailed in this section.

Uppercase represents fixed message text. Lowercase (or $$$$$$) represents variable text.

Message	Module	Severity
$$$$$$: ERROR IN MESSAGE <label> <12 octets of msg> **Explanation:** The session manager ($$$$$$ = SESMGR) or the connection manager ($$$$$$ = SESCON) received invalid protocol at the <label> subfield. The 'label' may be any of the following: INPLEN, LEVEL, NODATA, TYPE. **User Response:** There is a protocol problem between the two NetExes. Contact support personnel to investigate.	**NXSESS**	**7**
$$$$$$: ERROR IN PACKET <label>n **Explanation:** The session manager ($$$$$$ = SESMGR) or the connection manager ($$$$$$$ = SESCON) received an error on an internal NetEx call. The 'label' may be: NRBIND nnnn (unexpected NRB indicator value), or, NRBSTAT nnnn (NRB error status). **User Response:** If the problem persists, contact Network Executive Software support personnel to investigate.	**NXSESS**	**7**
ADAPTERS NOT CONFIGURED FROM NCT **Explanation:** NETEX$CONFIG is not available to NetEx. **User Response:** The CONEMANG output PAMFILE must be in a file known to NetEx as NETEX$CONFIG.	**NXINIT**	**14**
ALL OF THE NETEX ADAPTERS ARE HALTED. **Explanation:** A periodic check of the adapters has determined that there are none currently available. **User Response:** Attempt to re-start any necessary adapter(s) with the START ADAPTER command. If this fails look at the I/O error messages and, if necessary, contact the appropriate hardware adapter vendor's customer support to determine why the adapter(s) is/are failing.	**NXNETW**	**14**
APR FROM PATH:p localDref:remoteDref runid/remoteHost/[id] NREFS:local/remote **Explanation:** The connect failed or an active connection was interrupted. **User Response:** If the path should be working, investigate network problem or remote Netex issue.	**NXTRAN**	**10**
ASSIGN FAILURE OF nnnn FOR DREF aadd **Explanation:** Network attempt to assign a driver reference number failed. **User Response:** Check the status of the requested adapter (aa). It may be halted.	**NXNETW**	**14**

Message	Module	Severity
DMON: yymmdd hhmmss **Explanation:** Periodic time and date message issued in demand mode to keep the terminal remains active. **User Response:** None	**NXTRAN**	**14**
DNSLOOKUP: NO ADAPTER UP FOR DNS INPUT – SKIPPED **Explanation:** There are no adapters available to receive DNS lookup responses **User Response:** Start an adapter if possible, then LOAD NCT	**NXUTIL**	**15**
DNSLOOKUP: NO BUFFER FOR DNS LOOKUP - SKIPPED **Explanation:** Netex couldn't allocate a buffer to do the DNS lookup function **User Response:** Wait for a while, then LOAD NCT	**NXUTIL**	**15**
DRTIMER: TERMINATING BECAUSE IPC HEARTBEAT STOPPED **Explanation:** IPC has stopped updating the mailbox, perhaps it terminated **User Response:** Restart IPC and NTX		**15**
DRIN: IGNORED DNS RETURN FOR GNA: gggg **Explanation:** the ipaddr returned for this GNA is not used because a specific SET IP was done for it. **User Response:** None	**NXDCOM**	**4**
DRIN: GNA-IP TABLE FULL, CANNOT ADD NEW VALUE **Explanation:** the maximum number of GNAs are in the table already **User Response:** CLEAR GNA for any that are not needed	**NXDCOM**	**15**
DRIN: DNS RETURNED GNA: gggg IP ADDRess: a.b.c.d **Explanation:** DNS lookup returned the stated IP address for the stated GNA **User Response:** None	**NXDCOM**	**0 if IPaddr is not zero; otherwise 4**
DRTIMER: TERMINATING BECAUSE IPC HEARTBEAT STOPPED **Explanation:** IPC has stopped updating the mailbox, perhaps it terminated **User Response:** Restart IPC and NTX	**NXDCOM**	**15**
DRTIMER: TERMINATING BECAUSE IPC HEARTBEAT STOPPED **Explanation:** IPC has stopped updating the mailbox, perhaps it terminated **User Response:** Restart IPC and NTX	**NXDCOM**	**15**

Message	Module	Severity
DRVERR FROM ccc STATUS: sss FUNCTION: fff [UNIT: uuu] timestamp **Explanation:** error return to Netex driver from IPC **User Response:** None ***Function:*** • 01 init NTX • 02 term NTX • 03 status (reserved) • 04 Halt adapter • 05 Start adapter • 06 Set Message Filter Level • 07 Check License • 010 write C-format • 011 write A-format • 020 read message • 021 read data • 022 discard data ***Status:*** • 0 Normal return • 1 NTX# (PARM1) out of range • 2 NTX# already connected (Function=01) or not connected (Function=02) or not halted/started (04/05) • 8 License Expired, IPC will terminate. • 10 Init found one of the specified GNAs is already in use, the init is rejected • 128 License Warning (See Console message)	**NXDCOM**	**15**
DRVINIT: INITIALIZE WITH IPC SUCCESSFUL **Explanation:** IPC has accepted the Netex initialize function **User Response:** None	**NXDCOM**	**14**
DRVINIT: WAITING FOR H300IPC TO INITIALIZE MAILBOX **Explanation:** IPC has not started and set up the mailbox common bank yet. **User Response:** Start the IPC background job	**NXDCOM**	**14**
DWRITE ERROR ON MSG LOOPBACK **Explanation:** An HDL message was received, but in sending it back, an error was returned by the driver. **User Response:** A TRACE (MR) will show the associated driver error code.	**NXNETW**	**6**
ERROR: MUST CONFIGURE USE, LOGQ AND LOGF **Explanation; The operator entered a SWITCH command. The command was not executed. All three parameters: LOGQ, LOGF, and USE must be in the NTX$INITFILE when netex is started, before the switch command is available.**	**NXOPIF**	**-**

Message	Module	Severity
ERRNIT: OUT OF NITS OR SAVES **Explanation:** NetEx has run out of internal activity buffers (NITs). **User Response:** Add more NITs by increasing XNITS in the initialization file, or decrease requirements by running fewer concurrent sessions (SESMAX).	**NXMAIN**	**15**
HOST NUMBER IS OUT OF RANGE ON FIND BY HOST **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
HOST NUMBER WITHIN HOST GROUP IS OUT OF RANGE **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
INIT: DBANK TOO BIG. REDUCE MEMORY REQUIREMENTS **Explanation:** The end of the NetEx D-bank overlaps the start of the common D-bank. **User Response:** Reduce NetEx D-bank requirements by changing initialization parameters such as: SESMAX, DRIVMAX, XNITS, X256S,	**NXINIT**	**14**
INIT: ERROR DETECTED IN COMMON BANK - REMAP. **Explanation:** This message appears if the common bank is not mapped correctly. It is a fatal error. **User Response:** Remap the common bank correctly and re-run program	**NXINIT**	**15**
INIT: ERROR ON PARAMETER CARD: KEY=xxxxxx **Explanation:** An invalid parameter card exists in the initialization file. **User Response:** Correct the offending card.	**NXINIT**	**14**
INIT: HOST PARAMETER OMITTED-FATAL. **Explanation:** There is no HOST = hhhhhhhh parameter in the initialization file. **User Response:** Add the appropriate card.	**NXINIT**	**l5**
INIT: NETEX$TRACE IS NOT ASSIGNED **Explanation:** The user has requested tracing to disk or tape. The NETEX$TRACE file has not been assigned to this run. **User Response:** Add the appropriate @ASG statement	**NXINIT**	**l5**
INIT: ILLEGAL REMOTE CONSOLE CLASS, IGNORED **Explanation:** The ROPCLASS parameter value is invalid. **User Response:** Use A, C or G. The default is used until changed.	**NXINIT**	**14**
INTOPER: NO BUFFERS - INPUT IGNORED **Explanation:** There is not enough available buffer memory to process an operator keyin. **User Response:** Try again later.	**NXMAIN**	**14**

Message	Module	Severity
INTREDY: NO NIT ON NRBNITQ **Explanation:** In processing a user request, an NIT was lost. **User Response:** This is a fatal NetEx processing error. Submit dump to Network Executive Software.	**NXMAIN**	**15**
LENGTH FIELD IS INCORRECT WITH PAM SET **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
LOAD CURRENTLY ACTIVE **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
MEMGET: SUSPEND runid queue-name **Explanation:** An internal request for buffer space has been queued because there is not enough space available. **User Response:** When memory is freed, the request will be de-queued. If this condition occurs frequently, more buffer space should be configured. See BUFSIZE parameter.	**NXMAIN**	**8**
NCT IO ERROR STATUS = ss **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
NCT IS NOT FOR THIS HOST **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
NCT NOT LOADED YET **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
NCT OPERATION COMPLETED SUCCESSFULLY **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
NCT OR NCT-ENTRY TOO BIG - NEED nnnn BYTES **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
NETEX b.r yymm date time **Explanation:** NetEx sign-on message. The *b.r* is the base/release level. The *yymm* is the service level. The date and time is the current system date and time. **User Response:** None	**NXINIT**	**14**

Message	Module	Severity
NETEX INITIALIZED $$$$$ MSEC. DREND = $$$$$ CBEND = $$$$$ **Explanation:** NetEx initialization complete. Initialization time and ending addresses of the Common bank and D-bank are displayed. **User Response:** None	**NXINIT**	**14**
NETEX$CONFIG ASSIGN OR FREE FAILED STATUS sssssssssss **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
NO ENTRY IN THE NCT FOR THIS HOST **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**

Message	Module	Severity
NRECV: MESSAGE AND DATA DISCARDED $$$$$$ **Explanation:** The network layer discarded an incoming message for reason $$$$$$, which may be selected from the following. **BADCON Connect received had no PAM or dest was the connector.** **FROMAD PAM of incoming connect references unknown local adapter.** **LENGTH T-protocol length shorter than parsed segments.** **MAXQUE too many buffers in queue.** **N TYPE N-protocol subfield type invalid.** **NLEN N-protocol length shorter than parsed segments.** **NONREF Receiver's NREF incorrect.** **NOPRT1 Part 2 of a message received, but no part 1 is queued.** **NOSEGB unable to acquire a data segment buffer** **NSBLEN N-protocol subfield length greater than remaining N-length.** **NSTATE State of the N-connection does not allow this message type.** **ODLEN Odata received does not match length in the protocol.** **PAMEND Bad PAM - no end segment** **PAMERR Bad PAM (catch-all).** **PAMLEN Bad PAM length** **PAMTYP Bad PAM subfield type.** **PRTMIS Part 2 received does not match queued part 1.** **PTRUNC Pdata received is shorter than the length in the protocol.** **SESLEN S-protocol length invalid.** **T TYPE T-protocol subfield type invalid.** **TLEVEL Invalid T-protocol level.** **TRNREF Transmitter's NREF incorrect.** **710CNF Bad PAM link entry.** **User Response:** This may occur occasionally due to end-case conditions during a connect or disconnect. However, if it persists, obtain a message trace to investigate the reason.	**NXNETW**	**6**
NXCGY: CB FILE UNAVAILABLE - RELEASE EXCLUSIVE USE **Explanation:** The common bank file used by NetEx is in exclusive use and cannot be used to reload the NetEx common banks. **User Response:** Find out who has the exclusive assignment and release it.	**NXMAIN**	**15**
NXCGY: MCORES REQUEST FOR UNAVAILABLE STORAGE **Explanation:** Due to fragmentation of system memory, NetEx could not acquire the memory needed during initialization. **User Response:** Wait until a realtime program terminates, or configure less NetEx memory.	**NXMAIN**	**15**
NXCGY: NETEX ACCOUNT NOT ALLOWED TO GO REALTIME **Explanation:** The RUN account used by NetEx has not been given realtime privilege by the site administrator. **User Response:** Use an account with realtime privilege.	**NXMAIN**	**15**

Message	Module	Severity
NXCOMN: INTERFACE LEVEL IS INCORRECT. PLEASE RECOLLECT **Explanation:** The calling program was mapped with an incompatible version of the NetEx interface routines. **User Response:** Re-map program with the correct version of NXIFTN.	**NXCOMN**	**15**
NXTERM: END COMMON BANK TERMINATION **Explanation:** Informative message during NetEx termination. **User Response:** None	**NXCOMN**	**15**
NXTERM: START COMMON BANK TERMINATION **Explanation:** Informative message during NetEx termination. **User Response:** None	**NXCOMN**	**15**
OPERATION WAS ABORTED BY AN NCT RELOAD **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**
RQTERM runid TERMINATED **Explanation:** A session for the specified runid has terminated. **User Response:** None	**NXSESS**	**4**
SEMGR: UNABLE TO TOFFER NREF - STATUS nnnn **Explanation:** The session manager was unable to offer to the transport layer. **User Response:** None. If the situation persists, KILL NETEX and submit the dump.	**NXSESS**	**8**
SMGR NREF = nnnnn **Explanation:** The session manager is initialized with the given reference number. **User Response:** None	**NXSESS**	**4**
The $$$$$: SESSION nref runid appl-name date time **Explanation:** Informs of a session start or completion. $$$$$ may be SOFFR, SCONN or STERM. **User Response:** None	**NXSESS**	**4**
TRAN: MSG+ DATA DROPPED FOR NREF nnnn REASON $$$$$$ **Explanation:** The Transport layer has thrown away a received message. The REASON may be: NSTATE, TSTATE, or BADFLD. **User Response:** This may occur occasionally due to end-case conditions during connect or disconnect. However, if it persists, obtain a message trace to investigate the reason.	**NXTRAN**	**6**
UNABLE TO ACQUIRE A BUFFER **Explanation:** The following messages reside in NXUTIL, but are issued by NXINIT or NXOPIF in response to requests for information from the NCT.	**NXUTIL**	**14**

Message	Module	Severity
USER BUFFER OUT OF LIMITS **Explanation:** The NRB address submitted to NetEx is not valid. **User Response:** Correct the calling program.	**NXCOMN**	**15**
WDSCAN: *WARNING* NOT DISPATCHED > n SECONDS **Explanation:** The watchdog timer has not been given control for at least the indicated time. **User Response:** This is a warning that the system may be very busy and Netex performance may be affected.	**NXMAIN**	**8**
WDSCAN: STOP FLAG IS ON **Explanation:** The watchdog timer has detected that a flag has been set which requests NetEx termination. **User Response:** None. If the cause is unknown, submit the dump to Network Executive Software.	**NXMAIN**	**15**

Appendix D. IPC Messages

Message Format

IPC messages are displayed on the system console and/or the print file for the IPC run. IPC messages are described and suggested actions to be taken are detailed in this section.
Messages from IPC have the following format:

```
IPC <c> <ts> message_detail
```

Where:

<c> Indicates message class with:

F	= Fatal
E	= Error
W	= Warning
B	= Banner
I	= Informational
T	= Trace
D	= Debug

Any of these message class codes may have a "C" appended to indicate the message is a continuation of a preceding message of the same class.

<ts> Specifies the date and time of the message formatted as MMDD@HHMMSS.SSS

Fatal Messages

IPC F <*ts*> Non-octal NTX$INIT param NTXMAIL: <*value*>

The parameter NTXMAIL in initialization file "netex$config.NTX$INIT" must specify the BDI of the installed "NTXMBXD" AFCB for H300IPC. This BDI value must be specified in octal. Example:

```
NTXMAIL = 01111
```

IPC F <*ts*> Invalid NTX$INIT param NTXMAIL: <*value*>

The parameter NTXMAIL in initialization file "netex$config.NTX$INIT" must specify the BDI of the installed "NTXMBXD" AFCB for H300IPC.

IPC F <*ts*> NTX$INIT param NTXMAIL - Not Found

The parameter NTXMAIL could not be found in the "netex$config.NTX$INIT" initialization file for H300IPC.

IPC F <*ts*> main create_available_queue failed - no more info

The OS2200 system was unable to provide an available queue for Queue Banks at the PROGRAM_LOCAL level. (IPC uses Queue Banks to interface with CPCOMM.) The OS2200 system may be unusually busy. Try restarting IPC. If the problem persists contact support.

IPC F <*ts*> Read Mailbox Format Error NTX <*num*>

IPC has detected a structure error in the mailbox. Verify the correct NTXMBXD module has been copied into the SYSLIBH300IPC library.

IPC F <ts> Write Mailbox Format Error NTX <num>

IPC has detected a structure error in the mailbox. Verify the correct NTXMBXD module has been copied into the SYSLIBH300IPC library.

IPC F <*ts*> The H300IPC license is missing or invalid; IPC
IPC FC is not operational. For a new key contact support@netex.com

H300IPC is a licensed product. H300IPC was unable to find an unexpired key for this MCN and Siteid/Partition in the "netex$config.$KEY$" key file.

IPC F <*ts*> The H300IPC license has expired; IPC is not
IPC FC operational. For a new key contact support@netex.com

H300IPC is a licensed product. The current key for H300IPC has expired, and no key for H300IPC with a later expiration date was available in the "netex$config.$KEY$" key file.

IPC F <*ts*> CP_UDPLISTEN <*value*>

IPC uses the CP_UDPLISTEN function to received messages from CPCOMM. The function returned the provided error status value. These values are defined in the Unisys Communications Platform Programming Reference Manual.

IPC F <*ts*> cpCommReader errCode <*value*>

CPComm returned an error code on a read. Error code values are defined in the Unisys Communications Platform Programming Reference Manual.

IPC F <*ts*> Invalid CPCOMM_MODE param <*value*>

The CPCOMM_MODE parameter must be "A" through "H". This must match the installed value in CPCOMM. The default value is "A".

IPC F <*ts*> Invalid CPCOMM_PORT param <*value*>

The CPCOMM_PORT must be numeric. This is the port number NETEX will communicate on. The default value is 6950.

IPC F <*ts*> INVALID CPCOMM_PWD param <*value*>

The CPCOMM_PWD may contain characters "A" – "Z" or numbers "0" – "9". This password must match the value coded in your CPCOMM parameter file.

IPC F <*ts*> INVALID CPCOMM_UID param <*value*>

The CPCOMM_UID may contain characters "A" – "Z" or numbers "0" – "9". This userid must match the value coded in your CPCOMM parameter file.

IPC F <*ts*> INVALID IPCKEYIN param <*value*>

The IPCKEY may contain characters "A" – "Z" or numbers "0" – "9". This string will be the console keyin value used to communicate with IPC.

Error Messages

IPC E <*ts*> Error loading key:

An error occurred when attempting to load a license key for H300IPC. The specific failure is provided by one of the following three continuation messages:

IPC EC Unable to open KEY errno=

A key for H300IPC must be saved in a cataloged file with the name: "netex$config.$KEY$" Insure this file exists, is available to your IPC run, and contains your H300IPC key from Netex. errno is the value returned by the "C" open function.

IPC EC Unable to retrieve SYSINFO

A license key for H300IPC is tied to a specific system (MCN and Siteid/Partition). IPC must be able to retrieve this information from the platform. Try using the "SYSINFO" tool provided in the H300IPC release file. This utility uses the same interfaces as IPC and provides a crosscheck on the availability of this information. Additional status messages from the system calls may be provided in the IPC print file. If you require assistance from NetEx Software for this issue please send the IPC print file to NetEx Software Customer Support with your request for assistance.

IPC EC No valid key found for this system

H300IPC was unable to locate a key for this particular system (MCN and Siteid/Partition) in the "netex$config.$KEY$" file. Contact NetEx Software Customer Support (support@netex.com) for further assistance.

IPC E <*ts*> *CPCRDR Dropping bad QUEUE bank (TVA Failure)*

IPC received a queue bank from CPCOMM. CPCOMM passes the number of bytes in this queue bank. IPC uses a test virtual address to verify the access to the data. The read access was not valid for all the data. The queue bank is discarded..

IPC E <*ts*> *CPCrdr tossing Pkt len=nnn, NETEX len = nnnn*

IPC received a queue bank from CPCOMM. CPCOMM passes the number of bytes in this queue bank. The number of bytes returned from CPCOMM did not match the number of bytes that NETEX says should be in the packet. The packet is discarded...

IPC E <*ts*> sendConsole <*value*>

An error occurred when attempting to send a message to the system console. The status returned is displayed. The original message will may also be written to the IPC run print file if the message filter level is sufficiently high.

IPC E <*ts*> Unable to open NTX$INIT file errn0 = %d

IPC E <ts> Verify it is not FIELD DATA FORMAT

IPC was unable to open the "netex$config.NTX$INIT" initialization file. errno is the value returned by the "C" open function. Insure the file is cataloged and available to the IPC run. This file must be in quarter word format. (@ed,uq)

IPC E <*ts*> loadConvTbls Invalid data line: <*n*> in: NETEX$CONFIG.ebcidicascii/txt

IPC found invalid data in one of the conversion tables it was loading. The specific file element that was being read is displayed on the next line. The conversion table elements have 32 data lines plus any number of comment lines. Data lines must contain eight octal values separated by white space only. The eight values may be followed by a comment that begins with a period. Anything other than octal values and white space before a comment begins, or having less than eight values will cause this error. Compare your current conversion table elements with the valid ones provided by Netex in the H300IPC release file.

IPC E <*ts*> makeQueueHdr CREATE$QH status: <*value*>

IPC needs queue bank headers to pass queue banks between activities. The OS2200 system returned the error status displayed. The OS2200 system may be unusually busy, try starting IPC again. If the problem persists, contact NetEx Software Customer Support. The next message line provides additional status for Unisys support.

IPC EC makeQueueHdr CREATE$QH specific_status: <*value*>

This message follows the preceding message and provides additional status information which may be of use to Unisys support.

IPC E <*ts*> ntxAdapter binary DMODE malloc(<*bytes*>) failed

An error occurred when IPC attempted to malloc a data area. The number of bytes that were requested is displayed.

IPC E <*ts*> ntxWrite CP_TCPGETLIA status: <*value*>, for
IPC EC dest IP addr: <*value*>

CPCOMM returned the error status shown when IPC was asking which local IP address to use to reach a specific destination IP address. The destination IP address is provided in the second line.

IPC E <*ts*> ntxWrite CP_UDPQBSEND status: <*value*>

IPC received the error status displayed when attempting to send a UDP message through CPCOMM using Queue Banks via the CP_UDPQBSEND interface.

IPC E <*ts*> keyIn Failed status <*value*> flag <*value*>

IPC received the error status displayed when attempting to register a keyin with the EXEC. status and flag are the values returned by the EXEC on the keyin$$ ER.

IPC E <*ts*> keyIn input failed status <*value*>

IPC received the error status displayed when attempting to wait for operator input. Status is the value returned by the EXEC on the keyin$$ ER.

IPC E <*ts*> keyIn input not recognized

IPC received a keyin from the operator. It did not match any known commands. TERM, SWITCH and ABORT are currently the only recognized commands.

IPC E <*ts*> sendConsole (WTOR) <*value*>

IPC received an error status when attempting to write a message to the operator's console. The value was returned by the EXEC in response to the COM$ ER.

Warning Messages

IPC W <*ts*> The H300IPC license will expire on <*date*>

H300IPC is a licensed product. This message starts one month before the license expiration date. A write to operator message is also issued to insure the warning is noted. (Reply GO to the operator message).

IPC W <*ts*> The H300IPC license has expired; IPC will cease
IPC WC to operate on <*date*>. For a new key contact support@netex.com

H300IPC is a licensed product. The current license for H300IPC has expired, and no key for H300IPC with a later expiration date was available in the "netex$config.$KEY$" key file. The product will shut-down and be unable to be restarted after the date specified unless a new unexpired key for H300IPC is added to the "netex$config.$KEY$" file. A write to operator message is issued to insure the warning is noticed. (Reply GO to the operator message)

IPC W <*ts*> Key in KEY same as in memory - no change

A license reload request found that the H300IPC key in the "netex$config.$KEY$" file which had the latest expiration date, was the same as the key currently in use. Therefore no new key was loaded.

IPC W <*ts*> Key reload failed:

A license reload request resulted in a failure. The specific failure is provided by one of the following three continuation messages:

IPC WC Unable to open KEY.

A key for H300IPC must be saved in a cataloged file with the name: "netex$config.$KEY$" Insure this file exists, is available to your IPC run, and contains your H300IPC key from Netex.

IPC WC Unable to retrieve SYSINFO

A license key for H300IPC is tied to a specific system (MCN and Siteid/Partition). IPC must be able to retrieve this information from the platform. Try using the "SYSINFO" tool provided in the H300IPC release file. This utility uses the same interfaces as IPC and provides a crosscheck on the availability of this information. Additional status messages from the system calls may be provided in the IPC print file. If you require support from Netex for this issue please send the IPC print file to Netex with your request for support.

IPC WC No valid key found for this system

For H300IPC there was no key for this particular system (MCN and Siteid/Partition), nor was there any "Disaster Recovery" key, in the "netex$config.$KEY$" file. For a key contact NetEx Customer Support.

IPC W <*ts*> Only keys with expiration < current

A license reload was requested, but IPC found that the latest expiration date for any of the valid H300IPC license keys for this system, are for an earlier date than the expiration date of the key currently in use. The key file "netex$config.$KEY$" may have been reverted to an earlier cycle. This could also occur if the latest H300IPC key in this file was accidentally removed, replaced, or corrupted. A continuation message that will follow shows the new expiration that is being rejected, and the current expiration date which will be kept.

IPC W <*ts*> loadConvTbls Open failed for: errno=

IPC was unable to open one of the conversion table elements. The specific file and element that IPC attempted to open is displayed on the next line. Ensure that the specified file, or use name, is available to the IPC run and contains the specified element. Code conversion will not be available in H300IPC if this error occurs. errno is the value returned by the "C" open function..

IPC W <*ts*> loadConvTbls Excess data lines ignored in:

IPC found more than 32 data lines in one of the conversion tables it was loading. The specific file element that was being read is displayed on the next line. The conversion table elements have 32 data lines plus any number of comment lines. Compare your current conversion table elements with the valid ones provided by NetEx Software in the H300IPC release file.

IPC W <*ts*> cpCommReader queue is full, data lost:

Data is arriving faster than IPC and Netex can process it. IPC is discarding the data. Jobs will continue to process, but NetEx will need to request this data be retransmitted. This will significantly affect performance.

IPC YOU MUST CONFIGURE IPCUSE in NTX$INIT

IPC SWITCH COMMAND IS NOT ENABLED

The operator entered the IPC switch command. The IPCUSE parameter was not configured. This parameter must be in the NTX$INIT file before the switch command is allowed.

IPC SWITCH COMMAND COMPLETED

IPC has processed the SWITCH command and is now processing on the next cycle of the print file.

This is an informational message acknowledging the request to terminate IPC.

IPC W Please stop Netex NUM <value> :

In response to a request to TERM command, IPC found a copy of Netex is still active. The Netex with this number should be terminated. The IPC "TERM" command should be reissued.

IPC W Netex is down, enter <value1> KILL <value 2>

A term command was issued to IPC, The copy of Netex with a NTXNUM of value1 was found to be active. This copy of NetEx should be terminated. If the copy of Netex has in fact been terminated, You can correct the situation by entering the IPCKEY as <value 1>, and the NTXNUM as <value 2>. If this command is entered while that copy of netex is still active, it will be terminated.

IPC W or enter: <value1> ABORT to stop IPC and All copies of NetEx

By entering your IPCKEYIN <value1> ABORT, you are requesting IPC and all copies of Netex to abort.

IPC W KILL command is 6 characters in length

This command must be six characters in length. The word KILL followed by a space, followed by a single number. This number must be 0 – 5.

IPC W Netex Numbers are 0-5

The operator entered a KILL command to IPC and the number was not in the range. The command was not executed.

IPC W Operator killed Netex Num <value>

IPC is acknowledging the KILL command for the Netex Num displayed.

IPC W Netex <value> Killed

IPC successfully terminated the netex number listed.

IPC W Netex <value> not killed rc=%

IPC did not terminate the netex. The return code is displayed. This will most likely require IPC to be terminated and restarted in order to correct the situation.

IPC W GNA IN USE

During netex startup, IPC detected this NetEx was using a GNA address already in use but a second copy of NetEx. The start command fails

Banner Messages

IPC B <*ts*> IPC <*release level*> is starting

This message is produced when IPC is first started. It provides the specific release level of the IPC that is being started.

IPC B <*ts*> IPC is shutting down

IPC has received an operator request to terminate. IPC will terminate. Any copies of NetEx that are still attached, will terminate

IPC B <*ts*> IPC is responding to keyin <*value*>

IPC is displaying the operator keyin value for this run.

Informational Messages

IPC I <*ts*> Mailbox bank at: <*value*>

This message provides the Extended Mode virtual address of the Netex-IPC communication mailbox bank.

IPC I <*ts*> Console Filter Level set to: <*level*>

IPC received a request to set the filtering level for its console messages to the specified value. Only messages of a class level less than this filter level will be sent to the console. See: "SET IPCLOG/IPCONS Commands" for message class definitions.

IPC I <*ts*> cpCommReader errCode <*value*>
IPC IC Local IP = *xx.xx.xx.xx* Remote IP = *xx.xx.xx.xx*

IPC use the CP_TCPQBRCV function to receive Queue Banks from CPCOMM. The function returned the provided error status value. These values are defined in the Unisys Communications Platform Programming Reference Manual. The IP addresses displayed show the local and remote addresses involved in the error.

IPC I <*ts*> Message Filter Level set to: <*level*>

IPC received a request to set the filtering level for its normal output (PRINT$) to the specified value. Only messages of a class level less than this filter level will be sent to PRINT$. See: "SET IPCLOG/IPCONS Commands" for message class definitions.

IPC I <*ts*> New key: <*key*>
IPC IC New expiration: <*date*>

A license reload was successful and loaded the specified new key. The expiration date of the new key is displayed by the second line in YYYYMMDD format.

IPC I <*ts*> Attached to CPComm

IPC has successfully communicated with CPComm and completed the registration process.

IPC I <*ts*> cpComm listening on port <*value*>

IPC is listening for NetEx traffic on the specified port. The default value is 6950.

IPC I <*ts*> Operator entered a TERM command

The operator entered a term command to bring down IPC and all copies of NETEX using this IPC.

IPC I <*ts*> Requesting CPComm Mode <*value*>

IPC is attempting to communicate with CPComm using the mode specified. This value must match the mode value of your CPComm. The default is “A”.

IPC I <*ts*> Waiting for CPComm startup: <*value*>

IPC has attempted to communicate with CPCOMM. CPCOMM is not ready to process the request. Value is the error message returned by CPComm. It will be retried in 30 seconds.

Appendix E. Test Programs

NETEXEAT and NETEXGEN

The NETEXEAT and NETEXGEN programs on Unisys are compatible the NTXMGEN and NTXMEAT programs on Unix and zOS, if the "M" option (see below) is specified.

This pair of programs provides a one-way data transfer and timing test between systems, either intra-host (via the channel interface), or to/from the corresponding applications on the NetEx adapter. Many NetEx versions support compatible versions of these programs for heterogeneous NetEx testing. To use, first start the NETEXEAT program on the receiving system:

```
@XQT,<options> <Release Library>.NETEXEAT   on a Unisys host.
```

Option	Description
L	aligns prompt for telnet sessions - Required
M	requests the NTXMEAT offer be used else NETEXEAT will be offered
V	requests data validation

Next, execute the NETEXGEN program on the sending system:

```
@XQT,<options> <Release Library>.NETEXGEN   on a Unisys host.
```

Option	Description
L	aligns prompt for telnet sessions - Required
M	Connects to NTXMEAT offer. If not used connection to NETEXEAT
V	Requests data validation

The NETEXGEN program will prompt you for a string of parameters in fixed format, formatted like:

```
bbbbb ssssss llllll mmm odl hhhhhhhh ssnm
```

where:

bbbbb the number of blocks

ssssss the size of each block (in words)

llllll the number of loops

mmm the mode, in decimal (that is, 257 = octet mode = 000 hexadecimal)

If using the "V" option a mode value of 257 must be specified on the Unisys side. If NTXMGEN is running on the UNIX side, the DMODE mode must be specified as 0101.

odl Odata length (normally 000, see NRBPROTL description).

hhhhhhhh the hostname where NETEXEAT/NTXMEAT is offered

ssnm unused

After each completed loop, both programs will print elapsed time and bit rate.

Index