H210IPZ NetEx/IP®

for IBM z/OS Operating Systems

Release 7.4

Programming Reference Manual

MAN-API-H210IPZ-7.4

Revision Record

Revision	Description
01 (Sep. 2000)	Initial manual release by Network Executive Software Manual updated to correspond to release 5.0. Adds information for support of NetEx over IP networks.
02 (Apr 2001)	Manual updated to indicate Parallel channels are also supported on NESiGate device.
03 (Aug 2001)	Miscellaneous editorial changes. Manual prepared for electronic distribution.
04 (Jan 2002)	Manual updated to correspond to NetEx/IP release 6.0.
05 (Apr 2002)	Manual updated to correspond to NetEx/IP release 6.1.
06 (Jan 2009)	Manual updated to correspond to NetEx/IP release 7.0 Programming Reference Manual split from Software Reference Manual
7.1 (Feb 2011)	Updates for H210IPZ release 7.1 (version of manual to align with software release); refer to the Release Announcement for details of new features)
7.2 (03/2012)	Updates for H210IPZ release 7.2 (version of manual to align with software release); refer to the Release Announcement for details of new features). There are no updates to this manual other than to reflect the new release number.
7.3 (09/2014)	Manual updated to correspond to 7.3 release
7.4 (03/2015)	COBOL linkage added

Minor editorial revisions are not indicated.

You may submit written comments to:

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

Comments may also be submitted over the Internet by addressing e-mail to:

support@netex.com

or, by visiting our web site at:

http://www.netex.com

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

Preface

This manual describes how to use and install the H210IPZ NETwork EXecutive (NetEx®) software for the IBM z/OS operating systems. It also describes the general NetEx® program architecture, session concepts, and Application Program Interfaces (API's). Readers are not expected to be familiar with NetEx before using this manual. However, writing programs to the NetEx API's requires an understanding of programming in the supported languages; usage of NetEx requires an operational understanding of the host operating system.

The previous H210IP Release 6.1 Reference Manual was split into the following manuals for H210IPZ:

H210IPZ NetEx/IP for IBM z/OS Operating Systems Installation Reference Manual

H210IPZ NetEx/IP for IBM z/OS Operating Systems Operator Reference Manual

H210IPZ NetEx/IP for IBM z/OS Operating Systems Programming Reference Manual

The following manual had been previously split out and still remains for H210IPZ:

H210IPZ NetEx/IP for IBM z/OS Operating Systems Messages & Abend Codes Reference Manual

The *Operator Reference Manual* is intended for those responsible for daily operations of H210IPZ on z/OS.

The *Installation Reference Manual* is intended for those responsible for installing, configuring, and maintaining H210IPZ on z/OS

The *Programming Reference Manual* is intended for those responsible for using the NetEx/IP APIs, and contains the following information:

- "H210IPZ NetEx/IP Overview" introduces NetEx/IP and is intended for all readers.
- "Intertask Communication" explains the concepts of intertask communication using NetEx.
- "NetEx Request Block" explains the concepts of the NetEx Request Block.
- "Assembler Programming Interface" describes the assembler interface for a NetEx. programmer to communicate with NetEx, and to allow access to all.
- "C High Level Programming Interface" describes the C high level language program interface.
- "FORTRAN Programming Interface" describes the FORTRAN high level language program interface.
- "PASCAL Programming Interface" describes the PASCAL high level language program interface.
- "COBOL High Level Programming Interface" describes the COBOL high level language program interface.
- "Appendix A. Asynchronous Post Exit Processing Routine" describes how to specify the address of a user-written routine that receives control after the conclusion of a NetEx request.
- "Appendix B. Glossary" lists and describes commonly used terms and acronyms in this document.

Reference Material

Reference material may be found in the following publications:

Number	Title and Description
MAN-M&A-H210IPZ	H210IPZ NetEx/IP for IBM z/OS Operating Systems Messages and Abend Codes
MAN-OPR-H210IPZ	H210IPZ NetEx/IP for IBM z/OS Operating Systems Operator Reference Manual
MAN-INS-H210IPZ	H210IPZ NetEx/IP for IBM z/OS Operating Systems Installation Reference Manual
460195	H211 Bulk File Transfer (BFX™) Utility for IBM MVS Software Reference Manual
460201	H212R Print File Transfer (PFX™) Receiver Utility for IBM MVS Systems Installation and User Guide
460345	H212T Print File Transfer (PFX) Utility for IBM MVS Systems Soft-ware Reference Manual
MAN-REF-EFT213	eFT213 for IBM z/OS User Guide
MAN-REF-CASW	NESiGate NetEx/IP Host Channel-to-IP Gateway Reference Manual
MAN-CNET-CONFIG-MGR	"C" Configuration Manager and NetEx Alternate Path Retry (APR) User Guide

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 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™
International Business Machines Corp.	IBM, z/OS

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

Comments may be submitted over the Internet by addressing email to:

`support@netex.com`

or, by visiting our web site at:

`http://www.netex.com`

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

Document Conventions

The following notational conventions are used in this document.

Table 1. Documentation Conventions

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.
<u>value</u>	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.

Contents

Figures

Tables

H210IPZ NetEx/IP Overview

General

The NETwork Executive (NetEx®) software allows 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. All of these versions provide a common Application Programming Interface (API) to simplify programming and portability. Application programs are also available for use with NetEx, such as Bulk File Transfer (BFX™) which provides the capability to move bulk sequential file data from one computer system to another; Print File Transfer (PFX™) which automatically monitors and transfers print files from one system's print spool (queue) to another; and eFT/USER-Access®, which provides a powerful remote access, file transfer and scripting capability.

NetEx software resides as a subsystem within each IBM host involved in the communication. NetEx allows communications to take place at any time during host operations, independent of other functions in the system.

The following sections describe the characteristics of NetEx and how it uses the International Standards Organization (ISO) guidelines for open systems interconnection.

New Features

H210IPZ Release 7.3

Refer to the following manual for a description of the new features:

"*H210IPZ NetEx/IP for IBM z/OS Operating Systems Release 7.3 Installation Reference Manual*"

NetEx Characteristics

NetEx centralizes network considerations in a single piece of software. The following sections describe the characteristics of NetEx:

- External Interface
- Internal Interaction
- NetEx Connections
- Design Efficiency and Flexibility
- Block Segmenting
- Alternate Path Retry
- Remote Operator Interface
- User Exits
- Basic I/O Flow

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 FORTRAN, COBOL, C or Assembler 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 may be used to monitor and control certain NetEx functions.

Internal Interaction

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

To facilitate communication between hosts of different manufacture, NetEx supports character code conversion.

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.

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, and alternate path routing, and is able to tolerate long delays inherent in communication paths over satellite links.

NetEx optimizes data transfer throughput using a high degree of parallelism. That is, under normal circumstances, simultaneous 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 provides 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. This is especially useful for satellite communication.

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. For more information on APR, refer to the “C” Configuration Manager and NetEx Alternate Path Retry (APR) User Guide.

Remote Operator Interface

NetEx provides a remote operator interface that allows users to issue NetEx operator commands to other defined NetEx hosts on the network. Security features are provided.

User Exits

NetEx provides user exits at well-defined points for security, accounting, or other user-defined purposes. These exits are routines that are essentially do-nothing and may be replaced by user modules at installation time. Because there is a wide variety of user requirements, NetEx does not provide generalized security and accounting needs. User Exits can be implemented to satisfy these requirements.

Basic I/O Flow

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

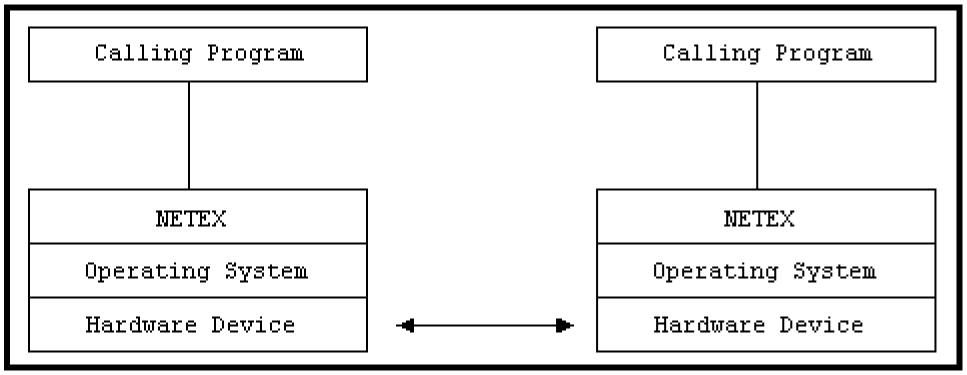


Figure 1. Basic I/O Flow

NetEx and the ISO Model

NetEx follows 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, and networks, 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 figure). By using this modular structure, the internal function of each layer is self-contained and does not affect the functioning of other layers.

Table 2. 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.

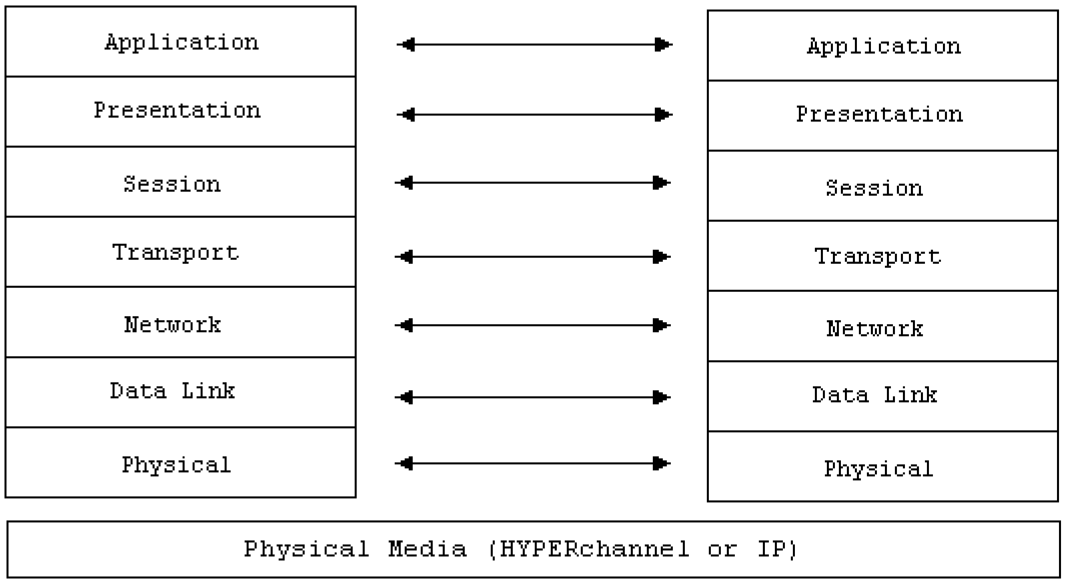


Figure 2. ISO Model Communication

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.

The figure below shows the correspondence between NetEx and application software to the ISO model. 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 applications provided by Network Executive Software.

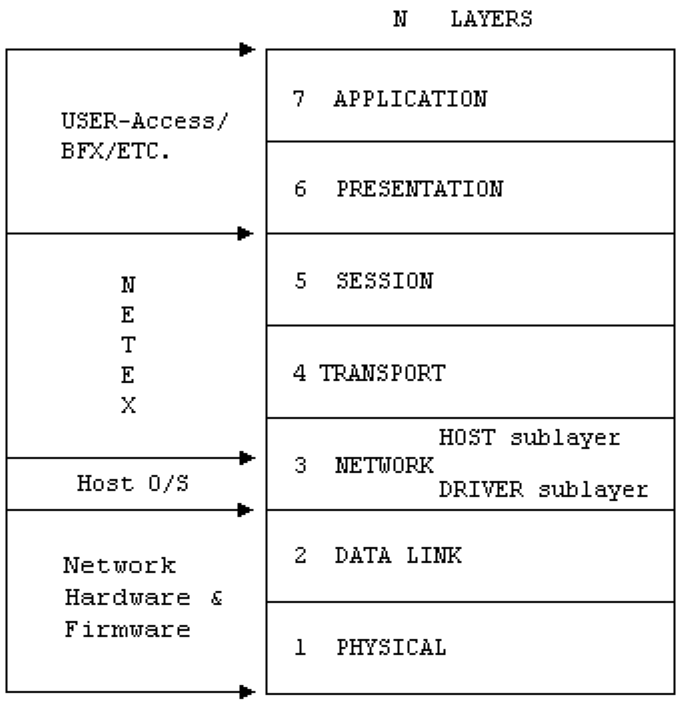


Figure 3. NetEx and the ISO Model

Session Layer Services

As the highest layer within NetEx (referring to the ISO model in Figure 2), 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 by using a standard NetEx Request Block (NRB) (containing parameters), and issuing the requests described in "NetEx Request Block". The session layer software processes 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 data is available. 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 NetEx/IP message, 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, and code conversion options, if these are provided by the user’s data mode parameter.

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
- Satellite 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 "NetEx Request Block" on page 41.

Session layer requests may be issued in Assembler, C, PASCAL, FORTRAN or COBOL.

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.

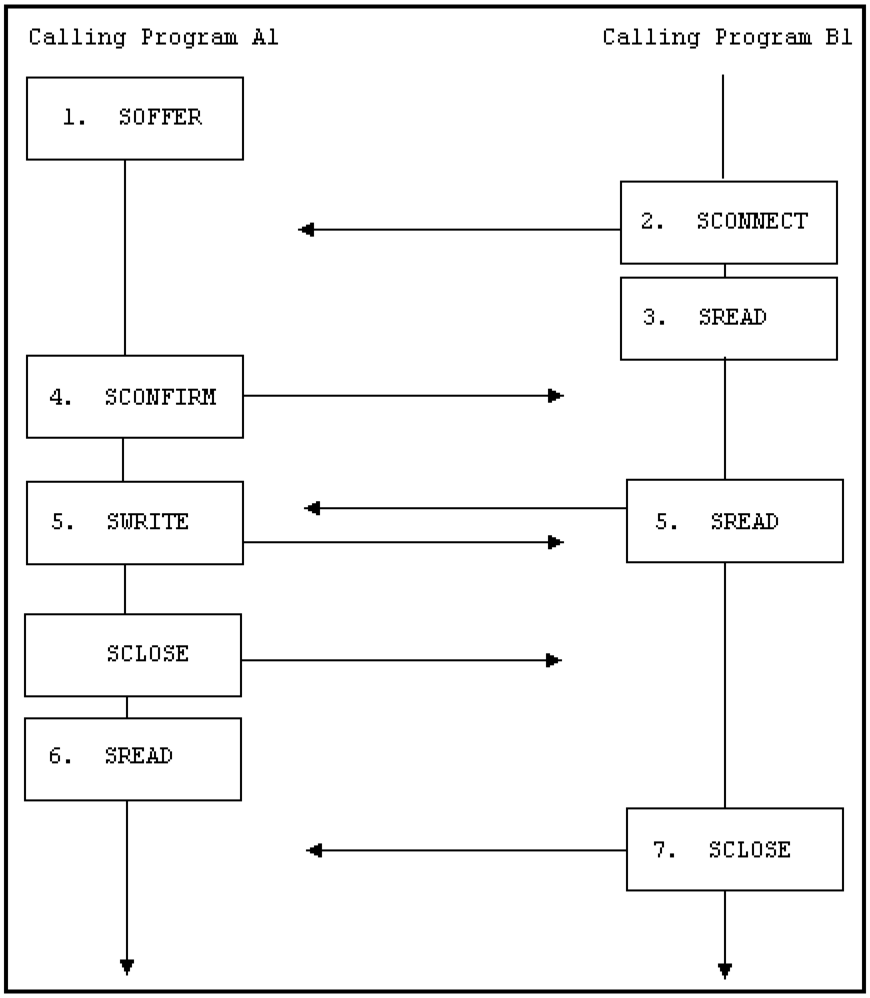


Figure 4. Two Programs Conducting a Session Using NetEx

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 "NetEx Request Block" on page 41) is a block of parameters used to signal requests to NetEx and to return status to programs.

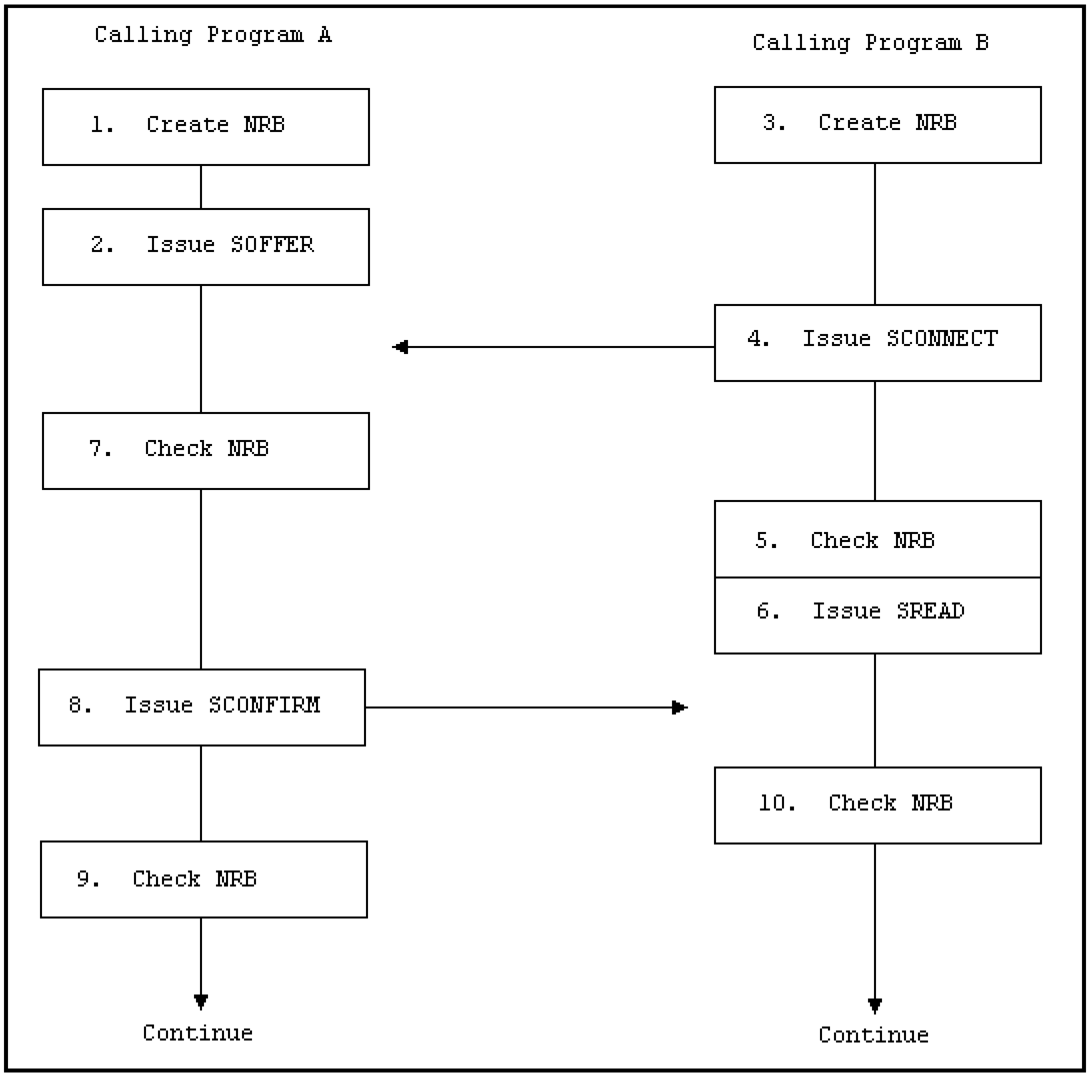


Figure 5. Establishing a Connection

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 "NetEx Request Block" on page 41.

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.

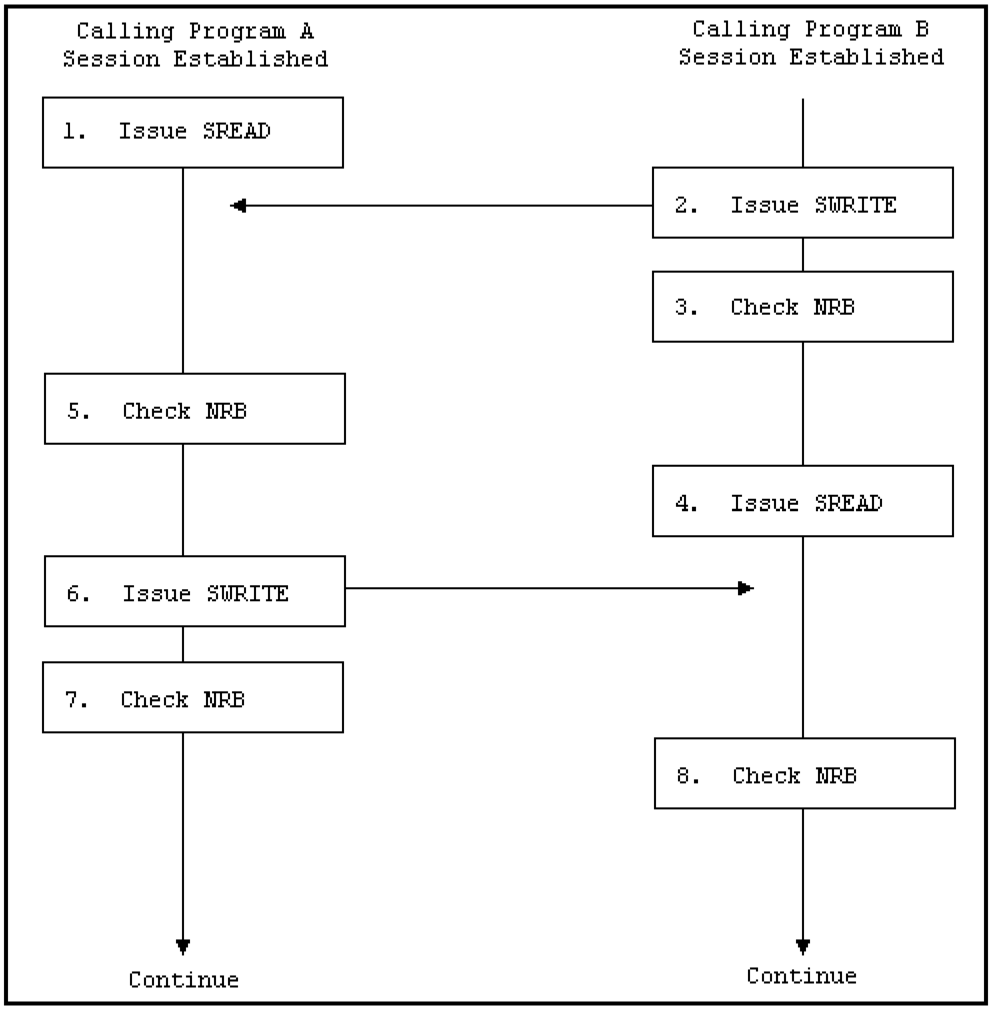


Figure 6. Write/Read Data Transfer

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 36.

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 satellite communication (see “Satellite Communication” on page 38 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 27) 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 35.

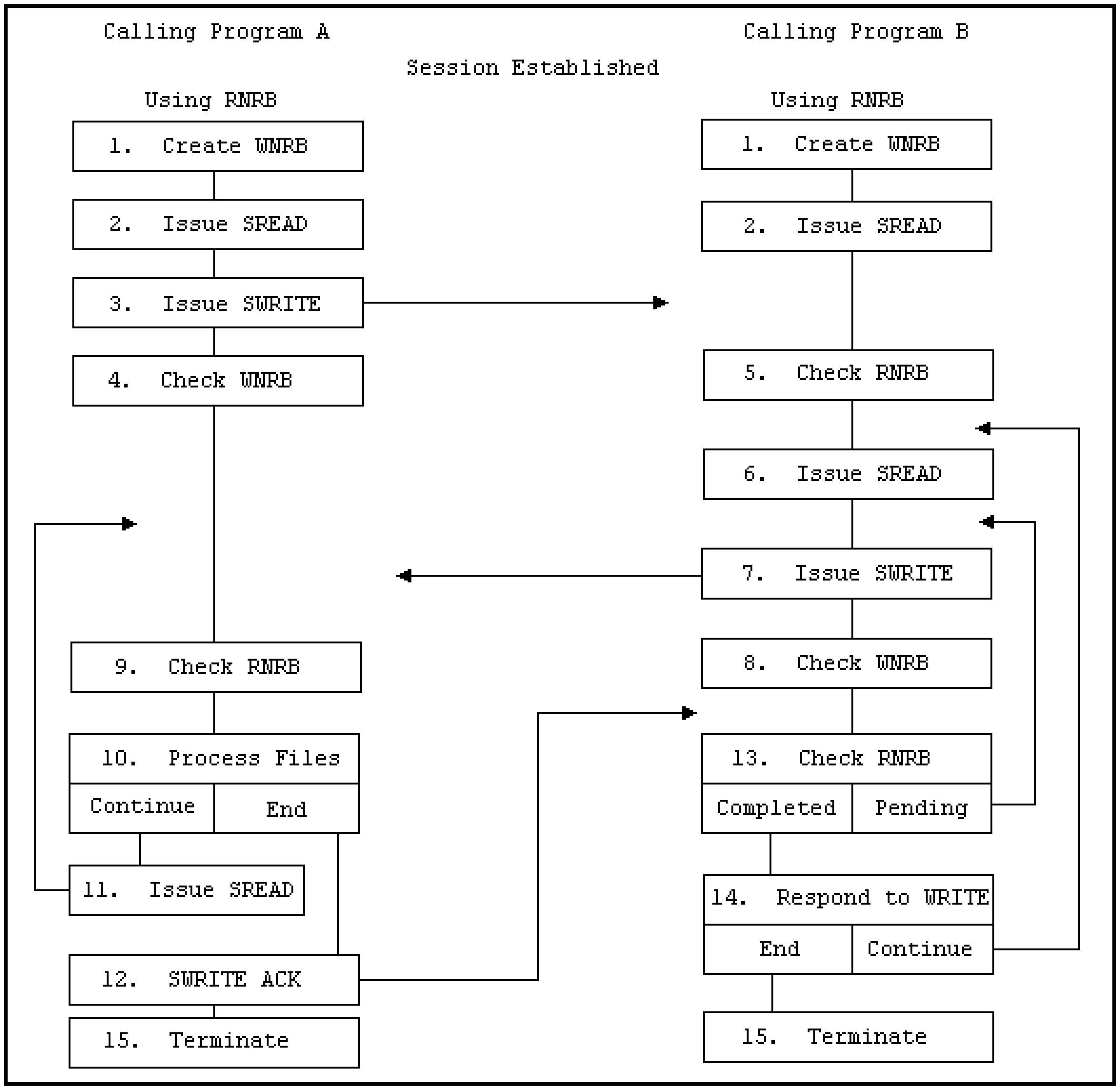


Figure 7. Concurrent SREAD and SWRITE Requests

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 35.

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 36.

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 27). 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.

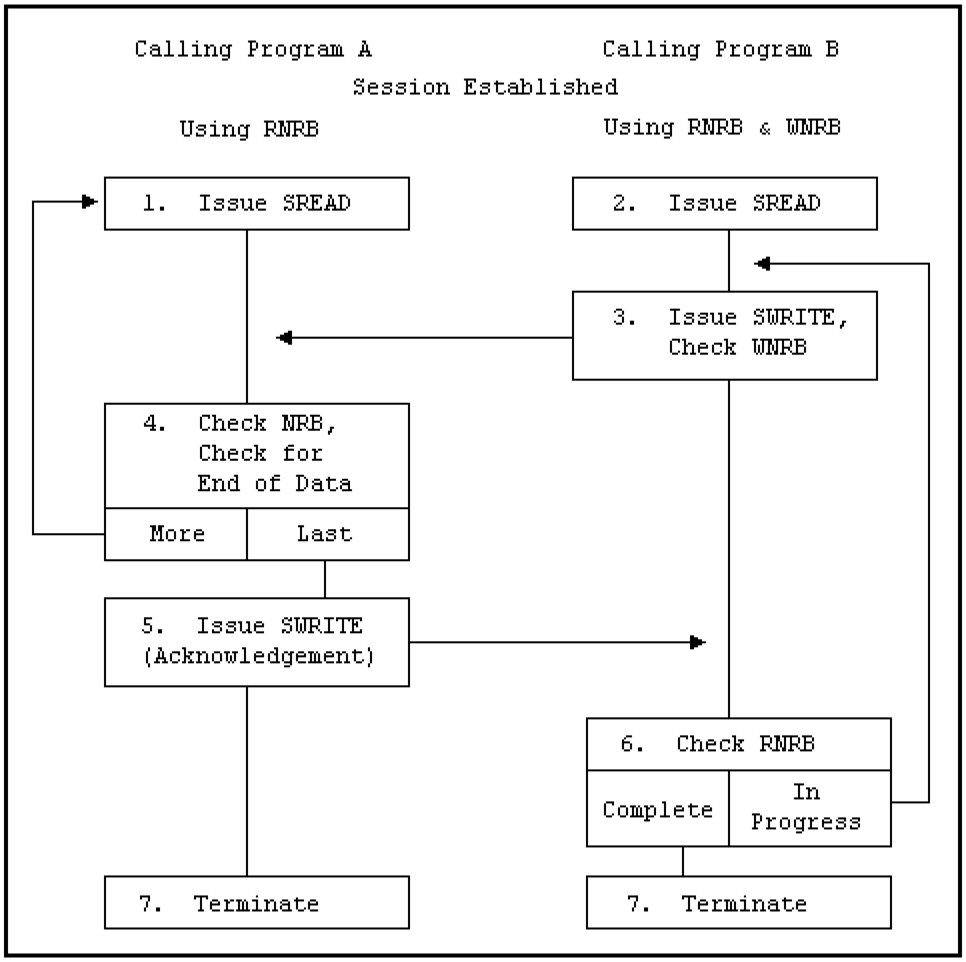


Figure 8. One-Way Data Transfer

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 35.

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.

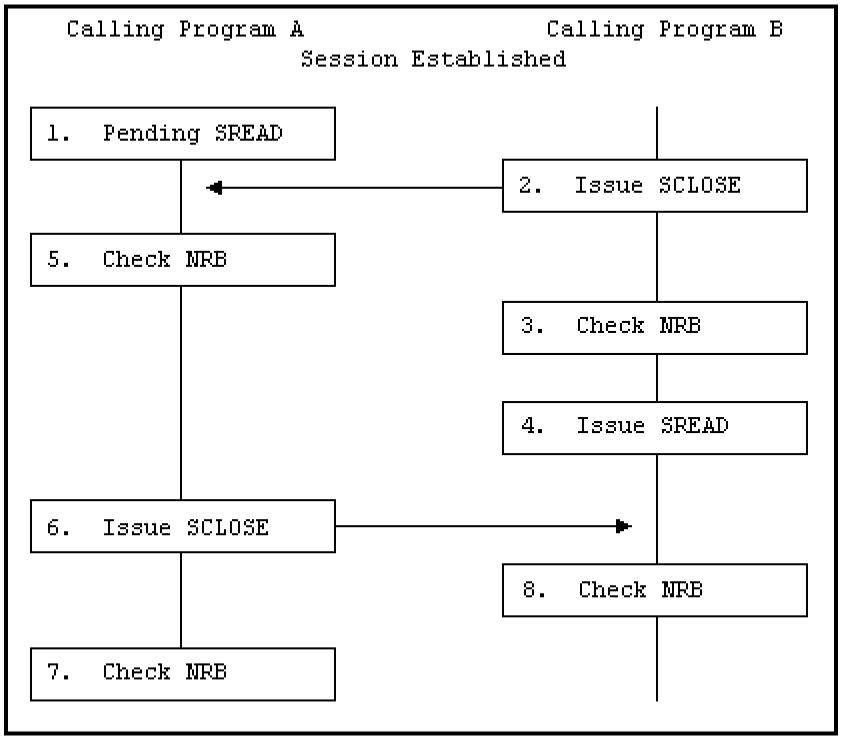


Figure 9. Normal Session Termination

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.

Satellite Communication

Satellite 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 the satellite subsystem (approximately 600 ms), 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 determines 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 describes 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 described in “Understanding NRBSTAT Error Codes” in the *H210IPZ NetEx/IP for IBM z/OS Operating Systems Messages and Abend 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 either StorageTek hardware or 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 adapter output and adapter input. The caller must determine which assembly/disassembly modes and code conversion tables are meaningful to the adapters. (Refer to the appropriate adapter hardware reference manual for a discussion of assembly/disassembly for that adapter.)

Automatic Code Conversion

The caller specifies only the source character set and the destination character set. NetEx uses available code conversion hardware, and uses software code conversion when necessary. Code conversion is done in module NXMCCV, which is a common module for both H210IP and H280.

NetEx Request Block

General

The NetEx Request Block (NRB) is an array of parameters that pass information between calling programs and NetEx. The NRB is the only means calling programs and NetEx communicate with one another. A calling program creates the NRB and updates the NRB using NetEx requests to pass information to NetEx, or NetEx may update the NRB to pass information to a calling 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 through the NRB. Therefore, only one NetEx session request may use an NRB at one time. If using concurrent READ and WRITE requests, or if connecting a server application to more than one program at a time, several NRBs must be used.

This section discusses the following topics:

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

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 words may be referenced by Assembler programs using the names shown in Figure 10 if using the NRBD macro (described in "Assembler Programming Interface" on page 53). 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 the status of the request issued by the user. If the request is currently in progress, the entire word contains a negative. If the request completed successfully, then NRBSTAT is 0. If NetEx or the service routine detects an error, the word contains an integer positive error code. See the "Understanding NRBSTAT Error Codes" section in the *H210IPZ NetEx/IP for IBM z/OS Operating Systems Messages and Abend Codes* for explanations of these error codes. The implementation user interface may then immediately examine the NRB results if a zero or positive value is found in NRBSTAT.

NRBIND

NRBIND indicates the type of data received in response to a READ or OFFER request. If any of those two read type requests are issued, NRBIND always receives a nonzero value.

The values returned in NRBIND are defined below:

(1)	**Connect indication**
(2)	**Confirm indication**
(3)	**Normal data indication**
(4)	**Reserved**
(5)	**Close indication**
(6)	**Disconnect indication**

If a write type request (WRITE, CONNECT, CONFIRM, or DISCONNECT) is issued, the returned value of NRBIND is usually zero. If an error is returned to the write type request, the connection is broken or was never established. A disconnect indication (6) is then set in NRBIND.

If an operation did not complete successfully, then NRBSTAT is set to a positive, nonzero value. If NRBSTAT is nonzero, then NRBIND has 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) is in NRBIND.
- If the error means that the request could not be processed, but the connection remains in effect, then NRBIND is set to zero.
- If the data is damaged in input (for example, the user buffer too small) then NRBIND reflects 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.

If a write type request (SWRITE, SCONNECT, SCONFIRM, or SDISCONNECT) is issued, then the caller must place the outgoing length information in NRBLEN and NRBUBIT. On a read type operation (SREAD or SOFFER), the NRBLEN and NRBUBIT fields are ignored at the time the request is issued. NetEx sets these fields to the length of the incoming data when the read type operation completes.

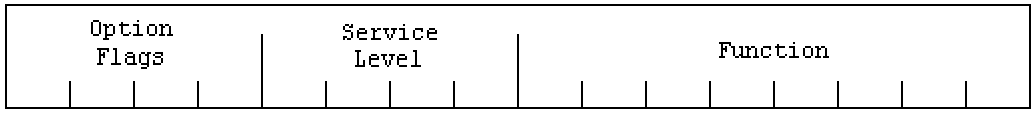
For example, CDC CYBER wants to send exactly 35 of its 60-bit CM words to an IBM processor and wants them returned at a later date. The CYBER user specifies NRBLEN=35 and NRBUBIT=0. Datamode is BIT STREAM. NetEx records that 35*60=2100 bits of information was sent over the network. The IBM user receives the information with NRBLEN=263 (bytes) and NRBUBIT=4 (bits). The IBM user can later specify the same length parameters on the output and return precisely 35 CM words back to the CYBER.

Note: Programs that do not use the NRBUBIT field may ignore its existence. The NRBLEN ensures that all information sent by the other machine is stored or processed.

NRBREQF, NRBOPSRV, NRBREQ

NRBREQF contains two unused bytes and two 1-byte fields that together constitute the request code that is to be given to NetEx.

NRBOPSRV is the third byte of NRBREQF; NRBREQ is the fourth byte of NRBREQF. NRBOPSRV and NRBREQ have the following format:



NRBOPSRV bits 0-3: Option Flags

Refers to optional processing that NetEx performs on the request. These flags are bit significant. The bits are assigned (in hexadecimal numbers) as listed below:

0
(No bit set) WAIT. NetEx or the NetEx user interface is not to return control to the user program until the request is complete

8
No option selected: normal processing

4
Reserved

2
Reserved

1
Reserved

NRBOPSRV bits 4-7: Service Level

Indicates if the request is a SESSION or other type of request. The values are assigned (in hexadecimal) as listed below:

0
Session request

3	Driver request
1-2	Reserved
4-F	Reserved

NRBREQ: Function

Indicates the specific type of request to be issued. The total request code is produced by combining the Function and Service Level. For example, SREAD is a 0 Service Level plus a 82 Function for a 082 (hexadecimal) request code. The values are assigned (in hexadecimal) as listed below.

01	Connect
02	Confirm
03	Write
04	Reserved
05	Close
06	Disconnect
07-80	Reserved
81	Offer
82	Read
83	Obtain Driver Status
84-FF	Reserved

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

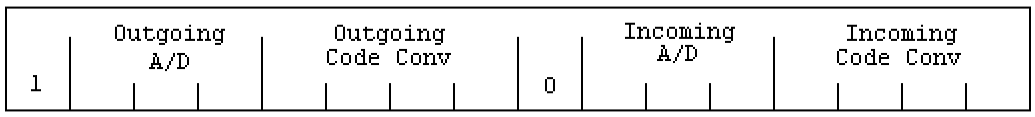
Specify NRBDMODE (datamode) when communicating to a non-IBM host. The transmitting NetEx program specifies NRBDMODE on either an SCONNECT, SCONFIRM, SWRITE, SCLOSE, or SDISCONNECT request. It is always specified as a 16-bit quantity. Datamode is forwarded through all layers of NetEx. When the receiving 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. This field is effectively unused since H210IP NetEx does not support hardware code conversion.

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. This field is effectively unused since H210IP NetEx does not support hardware code conversion.

0

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

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.

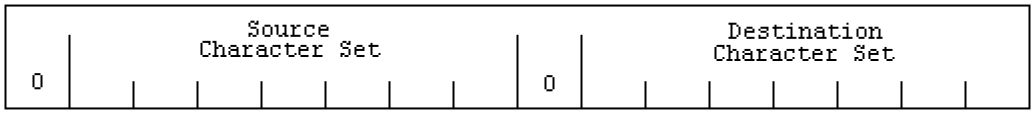
Octet

Eight-bit binary quantities are sent from one machine to another, using an 8-bit byte representation appropriate to each machine.

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 3) 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 3) 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 3. Auto Datamode Character Sets

Indicator	Conversion Option
0	Bit stream mode
1	Octet mode
2	ASCII (8 bit)
3	EBCDIC

Indicator	Conversion Option
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 “NetEx Request Block” on page 41 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

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 BLKI of one program is compared with the BLKO (output maximum buffer size) specified at the other end, and the lesser of the two values are returned in the two respective fields.

If the program is the CONNECTing one, then the negotiated results are returned in the NRB along with the CONFIRM data read following the CONNECT. The OFFERing program receives the negotiated values upon 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 two values controlling these buffer sizes. Refer to "Initialization Statements" in the *H210IPZ NetEx/IP for IBM z/OS Operating Systems Installation Reference Manual.*

- Specify the default input and output block sizes if these values are not specified (left zero) by the caller (DEFBI/DEFBO).
- Specify the maximum input and output block sizes permitted by the installation (MAXBI/MAXBO). Specifying a zero returns the default block size and a -1 returns the maximum block size.

Block Negotiating Process Example

Assume program A issues a CONNECT with BLKI=256 and BLKO=8192. The OFFERing program B that A connects to specifies zero, the default for both values. Installation defaults on the B CPU are BLKI=4096 and BLKO=4096. When the OFFER completes, B sees BLKO=256 and BLKI=4096, the minimum of the two sets of values. When A's READ following the CONNECT completes, it sees BLKI=256 and BLKO=4096 that are the same values as B with the directions reversed.

NRBPROTA

NRBPROTA contains the address of a buffer to receive data or that contains data to be processed as unconverted 8-bit binary quantities (user ODATA). This data is transmitted across the network as part of the message proper and is not subject to data assembly/disassembly or code conversion.

NRBPROTL

NRBPROTL contains the length of the buffer or the data contained within the buffer pointed to by NRBPROTA.

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.

NRBSSNM

NRBSSNM is a four-byte field word that specifies the IBM MVS subsystem name. This subsystem name uniquely identifies the NetEx on an MVS system that serves as the local host for an application.

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 Programming Interface

General

This section describes the following topics:

- Assembler Interface
- Macro Parameter Format
- NRB - Define NetEx Request Block Storage
- NRBD - Symbolically Define NetEx Request Block
- SOFFER Assembler Request
- SCONNECT Assembler Request
- SCONFIRM Assembler Request
- SREAD Assembler Request
- SWRITE Assembler Request
- SWAIT Assembler Request
- SCLOSE Assembler Request
- SDISCONN Assembler Request

Assembler Interface

The assembler interface for NetEx allows the assembler programmer to communicate with NetEx, and allows access to all NetEx facilities. The NetEx user interface consists of three parts:

- A macro to generate a NetEx Request Block in storage (NRB).
- A macro that generates a NRB DSECT (NRBD), allowing the programmer to access the fields of the NRB symbolically.
- A set of macros that correspond to the active NetEx requests. These macros can update fields in the NRB before NetEx is called, and they generate code to call the NetEx User Interface to perform NetEx services. The following macros are presented in the approximate order in which they are used:
    - SOFFER
    - SCONNECT
    - SCONFIRM
    - SREAD
    - SWRITE
    - SWAIT
    - SCLOSE
    - SDISCONN

Macro Parameter Format

The user-supplied portion of the parameters associated with the NetEx macro parameters are specified using the following conventions, unless otherwise noted:

exp

This expression fills a specified field of the NRB. The term exp indicates a supplied parameter. There are three types of supplied parameters:

expression

This parameter is treated as an RX-addressable expression. It may be a constant term (for example, 1234, X'200') or a symbolic expression (for example, BUFLEN, BUFEND-BUFSTART). Constants may have any 32-bit integer value. Symbolic expressions must have values less than 4096.

instruction:parameter

This parameter allows greater flexibility in parameter specification. Preceding a colon in the expression is an assembler op code for an instruction to place a parameter in a register (for example, L for LOAD). The target of the assembler instruction follows the colon. Examples are shown below:

```
L:BUFLEN
L:=F'10000'
LH:CHARMODE
```

(register)

This parameter specifies that the desired value is already in a general purpose register (enclosed in parenthesis). Registers 1 through 15 may be used for this purpose. Examples are shown below:

```
(2)
(R12)
(LENREG)
```

name

This specifies a string of eight alphanumeric characters sent to NetEx. This parameter can take one of two forms:

string

The first eight characters of the parameter are used as the string to be passed to NetEx. If the parameter is shorter than eight characters, it is padded with blanks. Examples are shown below:

```
HOSTA
NETEX
```

(register)

Specifies a register (enclosed in parenthesis) that contains the address of the eight-byte character string. Registers 2 through 15 may be used for this purpose. Examples are shown below:

```
(2)
(R12)
(LENREG)
```

NRB - Define NetEx Request Block Storage

The NRB macro defines storage containing the fields for a NetEx Request Block (NRB). All parameters that are normally supplied by the user for NetEx can be used in this macro.

Table 4. NRB Macro

Optional Name	**Operation**	**Optional Parameters**
LABEL	NRB	NAME=*NAME* ,DEST=*HOST* ,BLKI=*SIZE* ,BLKO=*SIZE* ,TIME=*SECONDS* ,AREA=*ADDRESS* ,LEN=*DATALENGTH* ,OAREA=*ADDRESS* ,OLEN=*DATALENGTH* ,UBIT=*BITCOUNT* ,MODE=*DATAMODE* ,MAXRT=*RATE* ,MAXRT=<u>0</u> ,SSNM=*NAME* ,SSNM=<u>NETX</u> ,CLASS=<u>0</u> ,CLASS=1 ,CLASS=2

NRB Parameters

The parameters described below may be supplied with the NRB macro. All parameters are optional. If no parameters are supplied, a blank NRB generates with entirely default or unspecified values (binary zeroes).

LABEL

Specifies a standard Assembly language macro label. This label may be used to reference the NRB when subsequent NetEx service calls are made.

NAME

Specifies the task name connected to on a remote host if a later SCONNECT call is issued against this NRB, or the name to be offered (OFFER) to NetEx if a subsequent SOFFER call is issued. This parameter is a one to eight character string. If omitted, this parameter must be supplied by modifying the NRB or using the NAME parameter in the SCONNECT or SOFFER macro.

DEST

Specifies the host processor name connected to if a later SCONNECT call is issued against this NRB. If this NRB offers (SOFFER) a connection, then this parameter is ignored. This parameter is a one to eight character string. If omitted, this parameter must be supplied by modifying the NRB or using the NAME parameter in the SCONNECT macro.

BLKI

Specifies the maximum size block (in addressable units or bytes for IBM) that the application is prepared to receive on an input operation. When the connection is established, this field contains the minimum of this application’s BLKI parameter and the remote application’s BLKO (output block size) parameter. The parameter should be an absolute expression. If omitted, it may be supplied by modifying the NRB or by specifying the BLKI parameter in the SCONNECT and SOFFER macros. If never explicitly specified, an installation default input block size is used.

BLKO

Specifies the maximum size block (in addressable units or bytes for IBM) that the application sends on an output operation. When the connection is established, this field contains the minimum of this application's BLKO parameter and the remote application's BLKI (input block size) parameter. The parameter should be an absolute expression. If omitted, it may be supplied by modifying the NRB or by specifying the BLKO parameter in the SCONNECT and SOFFER macros. If never explicitly specified, an installation default output block size is used.

TIME

Specifies the maximum number of seconds that a SOFFER or SREAD remains outstanding before the input type request completes abnormally. If left at zero (the default), the read type request never times out. The parameter should be an absolute expression. If omitted, it may be supplied by modifying the NRB or by specifying the TIME parameter in the SOFFER and SREAD macros.

AREA

Specifies the address of a data buffer used for NetEx I/O operations. The parameter should be an A-type address constant or expression. If omitted, this parameter may be supplied by the AREA parameters of all the NetEx service request macros.

LEN

Specifies both the maximum length of the data buffer (NRBBUFL) and the initial length to be written (NRBLEN) in addressable units (bytes for IBM). The parameter should be an absolute expression. If omitted, it may be supplied by the LEN parameter of all the NetEx service request macros.

OAREA

Specifies the address of a data buffer to be used for NetEx I/O operations. The parameter should be an A-type address constant or expression. If omitted, this parameter may be supplied by the OAREA parameters of all the NetEx service request macros.

OLEN

Specifies both the length of the data buffer addressable units (bytes for IBM). The parameter should be an absolute expression. If omitted, it may be supplied by the OLEN parameter of the NetEx service request macros.

UBIT

Specifies the unused bit count to be used on write type operations (SCONNECT, SWRITE, SCONFIRM, SDISCONN. The parameter should be an absolute expression. If omitted, it may be supplied by the UBIT parameter of the SCONNECT, SWRITE, SCONFIRM, and SDISCONN macros.

MODE

Specifies the DATAMODE to be used for write type operations. If omitted, the default is zero (bit stream data mode). The parameter, if supplied, should be an absolute expression. If omitted, it may be supplied at a later time by the MODE parameter of the SCONNECT, SWRITE, SCONFIRM, and SDISCONN macros.

MAXRT

Specifies the maximum data rate in thousands of bits per second used for this connection. If MAXRT=0 (the default) is specified, then NetEx constrains the connection throughput to the maximum speed of the media used to connect the two connecting applications. MAXRT may be later specified with the MAXRT parameters of the SCONNECT and SOFFER macros.

SSNM

Specifies the subsystem name of the NetEx to be used with this session. The default is NETX.

CLASS

Specifies the class of service to be inserted into the NRB.

CLASS=0

Specifies either (if both sides specify CLASS=0, type-2 protocol is used). The default is class 0 (recommended).

CLASS=1

Specifies type-1 protocol (used by release 1 NetEx products)

CLASS=2

Specifies type-2 protocol (used by release 2 and 3 NetEx products)

Usage Notes

Expansion of the NRB macro always generates 160 bytes of information, sufficient to contain a complete NetEx NRB. Note that none of the parameters of the NRB are required. All information needed to establish a connection may be supplied at a later time as parameters to the NetEx service request macros.

As noted in Understanding the NetEx Request Block, many NetEx applications requires two NRBs for the same connection so that NetEx reads and writes on the same connection may take place concurrently. To create a second NRB in assembler, define one NRB using the NRB macro or use the parameters in NetEx request macros. When the initial SCONNECT or SOFFER completes successfully, the entire working NRB should be copied to a blank NRB (perhaps a second NRB macro with no parameters). At this point, the copied NRB can be used for NetEx requests.

NRBD - Symbolically Define NetEx Request Block

This is the normal procedure for using NetEx: Issue a NetEx service call such as SREAD, wait for completion, and then examine the NRB directly to determine the status of the completed operation. The NRBD macro generates a complete DSECT so that the NRB can be addressed symbolically.

IMPORTANT
NetEx service request macro expansions use many of the symbols defined with the NRBD macro. The NRBD macro must be supplied in all assembler modules that issue NetEx service calls.

Optional Name	Operation	Parameters
LABEL	NRBD	

NRBD Parameters

LABEL

Specifies the name of the DSECT that defines the NRB. If omitted, the default is the DSECT name of NRB.

NRB DSECT

Figure 11 shows the expansion of the NRBD macro. The symbolic names used are consistent with the description of the fields given in the previous section.

Note: Fields marked reserved are not currently supported. These fields are not available to any caller. If you use these fields for any purpose, the results are unpredictable and constitute user error.

Figure 11. NRBD Expansion

```
NRB      DSECT
*
NRBBEG   DS   0F
NRBSTAT  DS   F                STATUS RETURNED FROM NETEX
NRBIND   DS   F                READ INDICATION
NRBCONIN EQU  1                CONNECT DATA READ
NRBCNFIN EQU  2                CONFIRM DATA READ
NRBNDTIN EQU  3                NORMAL DATA READ (FROM WRITE)
NRBEDTIN EQU  4                EXPEDITED DATA INDICATION (RESERVED)
NRBCLSIN EQU  5                CLOSE INDICATION
NRBDISIN EQU  6                DISCONNECT INDICATION
NRBSTSIN EQU  7                STATISTICS INDICATION (FROM NSTATISTICS)
NRBLEN   DS   F                LENGTH OF DATA
NRBUBIT  DS   F                UNUSED BIT COUNT
NRBREQF  DS   F                REQUEST CODE
NRBOPSRV EQU  NRBREQF+2,1      TOP NIBBLE&colon.OPTIONS; BOTTOM&colon.SERV
NREQSESS EQU  X'00'            SESSION   REQUEST
NREQTRAN EQU  X'01'            TRANSPORT REQUEST
NREQNETW EQU  X'02'            NETWORK   REQUEST
```

Figure 11. NRBD Expansion

```
NREQDRIV EQU  X'03'            DRIVER    REQUEST
NRBWEWAT EQU  X'00'            USER-INTERFACE WILL ISSUE WAIT (WAIT=Y)
NRBUWAIT EQU  X'80'            USER WILL DO WAIT (WAIT=N)
NRBUPOST EQU  X'40'            USER PROVIDED A POST EXIT
NRBREQ   EQU  NRBREQF+3,1      ACTUAL REQUEST CODE
NREQCONN EQU  1                CONNECT
NREQCONF EQU  2                CONFIRM
NREQWRIT EQU  3                WRITE
NREQEXWR EQU  4                EXPEDITED WRITE (CURRENTLY UNSUPPORTED)
NREQCLOS EQU  5                CLOSE
NREQDISC EQU  6                DISCONNECT
NREQSTAT EQU  7                STATISTICS
NREQASSN EQU  X'80'            ASSIGN
NREQOFFR EQU  X'81'            OFFER
NREQREAD EQU  X'82'            READ
NREQDSTA EQU  X'83'            OBTAIN DRIVER STATUS
NRBNREF  DS   F                NREF FOR THIS SESSION
NRBBUFA  DS   F                USER'S DATA BUFFER ADDRESS
NRBBUFL  DS   F                LENGTH OF THE BUFFER
NRBDMODE DS   F                ASSEMBLY/DISASSEMBLY MODE
NRBTIME  DS   F                TIMEOUT FOR A READ TYPE REQUEST
NRBCLASS DS   F                CLASS OF SERVICE
NRBMAXRT DS   F                MAXIMUM RATE OF TRANSMISSION
NRBBLKI  DS   F                BLOCK SIZE TO USE FOR INPUT
NRBBLKO  DS   F                BLOCK SIZE TO USE FOR OUTPUT
NRBPROTA DS   F                ADDRESS OF ODATA
NRBPROTL DS   F                LENGTH OF ODATA
NRBRESV1 DS   F                RESERVED
NRBRESV2 DS   F                RESERVED
NRBPNAME DS   CL8              LOGICAL PROCESS NAME (SOURCE)
NRBHNAME DS   CL8              LOGICAL NAME OF DESTINATION HOST
NRBRESV3 DS   F                RESERVED
NRBRESV4 DS   F                RESERVED
NRBUSIZE EQU   *NRBBEG             SIZE OF USER PORTION OF NRB
NRBOSDEP DS   0F               OS DEPENDENT SECTION
         ORG  NRBOSDEP+(2*4)
NRBECB   DS   F                ECB ADDRESS
NRBECBR  DS   F                REAL ECB
         ORG  NRBOSDEP+(10*4)
NRBSSNM  DS   F                SUBSYSTEM NAME
NRBFLAG1 DS   XL1
NRBUECB  EQU  X'80'            NEW MACRO EXPANSION WITH NTMECB
         ORG  NRBSTAT+(39*4)   ORG TO LAST WORD OF NRB
NRBASYNE DS   F                ASYNCHRONOUS POST ROUTINE
         ORG  NRBSTAT+(40*4)   FORCE SIZE TO 40 WORDS
NRBEND   DS   0F               END OF NRB EXPANSION
NRBSIZE  EQU  *NRBBEG         SIZE OF NRB
```

NRB
Common Name: NETEX Request Block

Macro ID: NRBD

DSECT name: NRB

Figure 11. NRBD Expansion

Created by: NRB
Size: 160 bytes (40 fullwords)

Function: Encodes user requests to NETEX and holds information returned by NETEX upon completion of a request.

```
          Offsets     Type    Length    Name      Function
  0  (0)             SIGNED     4      NRBSTAT    Status returned from NETEX

  4  (4)             SIGNED     4      NRBIND     Read indication
         .... .111                     NRBSTSIN   Statistics indication
         .... .11.                     NRBDISIN   Disconnect indication
         .... .1.1                     NRBCLSIN   Close indication
         .... .1..                     NRBEDTIN   Expedited data ind (reserved)
         .... ..11                     NRBNDTIN   Normal data read (from write)
          .... ..1.                    NRBCNFIN   Confirm data read
          .... ...1                    NRBCONIN   Connect data read

  8  (8)             SIGNED     4      NRBLEN     Length of data

 12  (C)             SIGNED     4      NRBUBIT    Unused bit count

 16 (10)             SIGNED     4      NRBREQF    Request code

 18 (12)             SIGNED     1      NRBOPSRV   Options; Service Level
          1... ....                    NRBUWAIT   User will do wait (WAIT=N)
          .1.. ....                    NRBUPOST   User provided a POST exit
          .... ....                    NRBWEWAT   User-interface will wait
                                                  (WAIT=Y)
          .... ..11                    NREQDRIV   Driver request
          .... ..1.                    NREQNETW   Network request
          .... ...1                    NREQTRAN   Transport request
          .... ....                    NREQSESS   Session request
 19 (13)             SIGNED     1      NRBREQ     Actual Request Code
          1... ..11                    NREQDSTA   Obtain Driver Status
          1... ..1.                    NREQREAD   Read
          1... ...1                    NREQOFFR   Offer
          1... ....                    NREQASSN   Assign
          .... .111                    NREQSTAT   Statistics (reserved)
          .... .11.                    NREQDISC   Disconnect
          .... .1.1                    NREQCLOS   Close
          .... .1..                    NREQEXWR   Expedited write (reserved)
          .... ..11                    NREQWRIT   Write
          .... ..1.                    NREQCONF   Confirm
          .... ...1                    NREQCONN   Connect

 20 (14)             SIGNED     4      NRBNREF    NREF for this session

 24 (18)             SIGNED     4      NRBBUFA    User's data buffer address

 28 (1C)             SIGNED     4      NRBBUFL    Length of the buffer

 32 (20)             SIGNED     4      NRBDMODE   Assembly/disassembly mode
```

Figure 11. NRBD Expansion

36 (24)	SIGNED	4	NRBTIME	Timeout for a read type request
40 (28)	SIGNED	4	NRBCLASS	Type of protocol
44 (2C)	SIGNED	4	NRBMAXRT	Maximum rate of transmission
48 (30)	SIGNED	4	NRBBLKI	Block size to use for input
52 (34)	SIGNED	4	NRBBLKO	Block size to use for output
56 (38)	SIGNED	4	NRBPROTA	Address of user ODATA
60 (3C)	SIGNED	4	NRBPROTL buffer	Length of user ODATA/ODATA
64 (40)	SIGNED	4	NRBRESV1	Reserved
68 (44)	SIGNED	4	NRBRESV2	Reserved
72 (48)	CHARACTER	8	NRBPNAME	Logical Process name (Source)
80 (50)	CHARACTER	8	NRBHNAME	Logical name of destination host
88 (58)	SIGNED	4	NRBRESV3	Reserved
92 (5C)	SIGNED	4	NRBRESV4	Reserved
96 (60)	EQUATE		NRBUSIZE	Size of user portion of NRB
160 (A0)	EQUATE		NRBSIZE	Size of NRB

SOFFER Assembler Request

The SOFFER macro allows a program to post its availability so it can accept a connection from a caller on the network.

Before issuing an SOFFER request, the user must provide an NRB (described in the “NetEx Request Block” section on page 41) to be used by the user interface. NetEx should be active in the local host.

Optional Name	Operation	Required Parameter	Optional Parameters

`LABEL`	`SOFFER`	`NRB=ADDRESS`	`,NAME=NAME`
			`,BLKI=SIZE`
			`,BLKO=SIZE`
			`,TIME=SECONDS`
			`,AREA=ADDRESS`
			`,LEN=BUFLENGTH`
			`,OAREA=ADDRESS`
			`,OLEN=BUFLENGTH`
			`,WAIT=Y`
			`,WAIT=N`
			`,WAIT=ADDRESS`
			`,MAXRT=RATE`
			`,SSNM=NAME`
			`,SSNM=NETX`
			`,CLASS=0`
			`,CLASS=1`
			`,CLASS=2`
			`,ECB=ADDRESS`

SOFFER Parameters

NRB

Specifies the address of the NRB data area to be passed to NetEx. The NRB and the creation of an NRB are discussed in “NetEx Request Block” on page 41.

The following parameters may be specified in any order.

LABEL

Specifies a standard Assembly language macro label.

NAME

Specifies the logical name of the offering user. The name may be up to 8 characters long. Although this parameter is not required in the SOFFER macro request, the name to be offered (OFFER) must have been supplied through a NRB macro or the SOFFER macro for a valid OFFER to occur.

BLKI

Specifies the maximum size of data blocks (in addressable units or bytes for IBM) that the user wants to receive during the lifetime of the session. This parameter is negotiated with the connecting application. The default is installation dependent.

BLKO

Specifies the maximum size of data blocks (in addressable units or bytes for IBM) that the user wants to transmit. This parameter is negotiated with the connecting application. The default is installation dependent.

TIME

Specifies the length of time in seconds that the SOFFER remains outstanding. A zero indicates no time limit. If not specified, the default is the last value used (the one left in the NRB).

AREA

Specifies the starting address of the area to receive data from a connect request. For an SOFFER to succeed, a valid buffer address must be supplied through the NRB macro or the SOFFER macro.

LEN

Specifies the length of the buffer used to hold data sent by the other application's SCONNECT request.

OAREA

Specifies the starting address of the area to receive user ODATA sent as part of the message proper from a connect request. For an SOFFER to succeed, a valid buffer address must have been supplied through the NRB macro or the SOFFER macro.

OLEN

Specifies the length of the buffer used to receive user ODATA sent by the other application's SCONNECT request.

WAIT

This is an optional parameter.

Y

Indicates the user interface routine executes a WAIT (described in "Assembler Programming Interface" on page 53).

N

Indicates the program regains control as soon as NetEx has accepted the request.

If **N** is selected, the program may issue a SWAIT on this NRB or a series of NRBs at a later time, or do other work and occasionally check to see if the request has completed.

ADDRESS

This is an exp-type value (an address, reg, or L:addr type constant), the address of an asynchronous NRB post exit. For more information, refer to the following SWAIT request description.

If this parameter is omitted, **WAIT=Y** is the default.

MAXRT

Specifies the maximum transmission rate in thousands of bits per second desired for the session.

SSNM

Specifies the subsystem name of the NetEx to be used with this session. The default is NETX.

CLASS

Specifies the class of service requested.

CLASS=0

NetEx specifies the class as either type 1 or type 2 protocol (if both sides specify CLASS=0, type-2 protocol is used). The default is class 0 (recommended).

CLASS=1

NetEx specifies type-1 protocol (used by release 1 NetEx products).

CLASS=2

NetEx specifies type-2 protocol (used by release 2 and 3 NetEx products).

ECB

This optional address is an exp-type value (an address, reg, or L:addr type constant), the address of a posted ECB when this request has completed. The address of the specified ECB are placed in the NRB for this request. The ECB parameter can be used with WAIT=Y, N, or ADDRESS.

If WAIT=N or ADDRESS use an MVS WAIT macro to wait for this ECB, the ECB should be cleared before it is ever specified on a request. It should be cleared again after a wait on it has completed. If a user program may also post the same ECB, a check must be done to ensure that the ECB is not double posted.

SOFFER Entry Parameters

The following NRB fields are used by SOFFER on entry.

BUFA
address for incoming data

BUFL
length of buffer to hold data

TIME
number of seconds offer outstanding

CLASS
class of service requested

BLKO
maximum transmission size acceptable

BLKI
maximum reception size acceptable

PROTA
buffer address for incoming user ODATA

PROTL
maximum length of the buffer at the address specified in NRBPROTA

ECB
address of the ECB to be posted (POST)

MAXRT
limit on transmission speed

PNAME
application name to offer

SSNM
subsystem name used to interface to NetEx

SOFFER Results

The following NRB fields are updated when SOFFER completes.

STAT
success/failure code

IND
contains connect indication

LEN

length of incoming data

UBIT
unused bit count of data

NREF
SREF assigned this connection

CLASS
class of service assigned

BLKO
maximum transmission data size

BLKI
maximum reception data size

PROTL
length of the user ODATA received into the buffer

MAXRT
maximum transmission speed of path

HNAME
name of host where SCONN originated

PNAME
name of application where SCONN originated

SSNM
subsystem name of the local NetEx

SCONNECT Assembler Request

The SCONNECT macro allows a program to request a session with a program that has previously issued an SOFFER.

Before issuing the SCONNECT request, an NRB must be provided for use by the user interface.

Optional Name	Operation	Required Parameter	Optional Parameters
LABEL	SCONNECT	NRB=*ADDRESS*	,DEST=*HOST* ,NAME=*NAME* ,BLKI=*SIZE* ,BLKO=*SIZE* ,AREA=*ADDRESS* ,LEN=*DATALENGTH* ,OAREA=*ADDRESS* ,OLEN=*DATALENGTH* ,MODE=*MODE* ,WAIT=Y ,WAIT=N ,WAIT=*ADDRESS* ,MAXRT=*RATE* ,SSNM=*NAME* ,SSNM=NETX ,CLASS=0 ,CLASS=1 ,CLASS=2 ,ECB=*ADDRESS*

SCONNECT Parameters

LABEL

Specifies a standard Assembly language macro label.

NRB

Specifies the address of the NRB data area to be passed to NetEx.

The following parameters may be specified in any order.

DEST

Specifies the logical name of the host computer that contains the application that has issued the corresponding SOFFER. The logical names of the host computers are installation dependent. This parameter is a one to eight character string. Although this parameter is not required for the SCONNECT request, a valid name must be supplied in the NRB macro or the SCONNECT macro for a connection to take place.

NAME

Specifies the logical name that the corresponding application used for its SOFFER request. This parameter is a one to eight character string. Although this parameter is not required for the SCONNECT request, a valid name must be supplied in the NRB macro or the SCONNECT macro for a connection to take place.

BLKI

Specifies the maximum size of data blocks (in addressable units or bytes for IBM) that the connector wishes to receive. This parameter is negotiated with the offering application. The default is installation dependent.

BLKO

Specifies the maximum size of data blocks (in addressable units or bytes for IBM) that the connector wishes to transmit. This parameter is negotiated with the offering application. The default is installation dependent.

AREA

Specifies the address of a buffer of data to be sent with the connect request. The data set is placed in the buffer associated with the remote application's SOFFER request. When the SOFFER completes successfully, the remote application has a connect indication (NRBCONIN) in NRBIND. Although AREA is optional, it must be specified as a valid, addressable memory location in either the SCONNECT or NRB macro.

LEN

Specifies the length of data (in addressable units or bytes for IBM) passed with the CONNECT request. The length (in addressable units) may be from zero up to 4K or ROOTSGSZ, whichever is less.

OAREA

Specifies the starting address of the area containing user ODATA to be sent as part of the message proper from a CONNECT request.

OLEN

Specifies the length of the data contained in the buffer specified by the OAREA parameter.

MODE

Specifies the assembly/disassembly modes used at both the transmitting and receiving adapters to deliver intelligible data to the receiving user. Refer to the datamode discussion in "NRBDMODE" on page 46.

WAIT

This is an optional parameter.

Y

This value indicates the user interface routine executes a WAIT (described in "Intertask Communication" on page 23).

N

This value indicates the program regains control as soon as NetEx has accepted the request.

If N is selected, the program may issue a SWAIT on this NRB or a series of NRBs at a later time, or do other work and occasionally check to see if the request has completed.

ADDRESS

This is an exp-type value (an address, reg, or L:addr type constant), the address of an asynchronous NRB post exit. For more information, refer to the following SWAIT request description.

If this parameter is omitted, WAIT=Y is the default.

MAXRT

Specifies the maximum transmission rate in thousands of bits per second desired for the session.

SSNM

Specifies the subsystem name of the NetEx to be used with this session. The default is NETX.

CLASS

Specifies the class of service requested.

CLASS=0

NetEx specifies the class as either type 1 or type 2 protocol (if both sides specify CLASS=0, type-2 protocol is used). The default is class 0 (recommended).

CLASS=1

NetEx specifies type-1 protocol (used by release 1 NetEx products).

CLASS=2

NetEx specifies type-2 protocol (used by release 2 and 3 NetEx products).

ECB

This optional address is an exp-type value (an address, reg, or L:addr type constant), the address of a posted ECB when this request has completed. The address of the specified ECB is placed in the NRB for this request. The ECB parameter can be used with WAIT=Y, N, or ADDRESS.

If WAIT=N or ADDRESS use an MVS WAIT macro to wait for this ECB, the ECB should be cleared before it is ever specified on a request. It should be cleared again after a wait on it has completed. If a user program may also post the same ECB, a check must be done to ensure that the ECB is not double posted.

SCONNECT Entry Parameters

The following NRB fields are used by SCONNECT on entry.

BUFA

address of outgoing data

LEN

length of outgoing data

PROTA

buffer address for outgoing user ODATA

PROTL

length of the data contained in the buffer at the address specified in NRBPROTA

ECB

address of the ECB to be posted (POST)

UBIT

data unused bit count

DMODE

datamode of data

CLASS

class of service requested

BLKO

maximum transmission size acceptable

BLKI

maximum reception size acceptable

MAXRT
limit on transmission speed

HNAME
alphanumeric host name

PNAME
alphanumeric application name

SSNM
subsystem name used to interface to NetEx

SCONNECT Results

The following NRB fields are updated when SCONNECT completes.

STAT
success/failure code

NREF
SREF (Session ID) assigned

LEN
set to zero

PROTL
set to zero

CLASS
class of service assigned

BLKO
maximum transmission data size

BLKI
maximum reception data size

MAXRT
maximum transmission speed of path

SSNM
subsystem name of the local NetEx

SCONFIRM Assembler Request

The SCONFIRM macro allows an offering program to confirm (to the connector) that the connection has been made and is the block size, class, and any optional data acceptable to the user. A negative response is a SDISCONN (disconnect).

Before issuing the SCONFIRM request, an SOFFER must complete successfully with an connect indication. The user must provide a NRB containing the information supplied by the previous SOFFER.

Optional Name	Operation	Required Parameters	Optional Parameters
LABEL	SCONFIRM	NRB=*ADDRESS*	,AREA=*ADDRESS* ,LEN=*DATALENGTH* ,OAREA=*ADDRESS* ,OLEN=*BUFLENGTH* ,MODE=*MODE* ,WAIT=Y ,WAIT=N ,WAIT=*ADDRESS* ,ECB=*ADDRESS*

SCONFIRM Parameters

LABEL

Specifies a standard Assembly language macro label.

NRB

This required expression specifies the address of the NRB data area to be passed to NetEx.

The following parameters may be specified in any order.

AREA

Specifies the address of a buffer of data to be sent to the corresponding application. The corresponding application receives this data with an SREAD macro and a confirm indication (NRBCNFIN) in the NRBIND field.

LEN

Specifies the length of optional data (in addressable units or bytes for IBM) to be sent with the request. The length may range from zero up to the lesser of the local host limit, the remote host limit, or the maximum allowed on the path. Although the LEN and AREA parameters may be omitted, their omission causes the values specified during the previous request handled by the NRB to be used.

OAREA

Specifies the starting address of the area containing user ODATA to be sent as part of the message proper from a CONFIRM request.

OLEN

Specifies the length of the data contained in the buffer specified by the OAREA parameter.

MODE

Specifies the assembly/disassembly modes used at both the transmitting and receiving adapters to deliver intelligible data to the receiving user. If omitted, the MODE specified during the previous use of the NRB is used. Refer to the datamode discussion under in "NRBDMODE" on page 46.

WAIT

This is an optional parameter.

Y

This value indicates the user interface routine executes a WAIT (described in "Intertask Communication" on page 23).

N

This value indicates the program regains control as soon as NetEx has accepted the request.

If N is selected, the program may issue a SWAIT on this NRB or a series of NRBs at a later time, or do other work and occasionally check to see if the request has completed.

ADDRESS

This is an exp-type value (an address, reg, or L:addr type constant), the address of an asynchronous NRB post exit. For more information, refer to the following SWAIT request description.

If this parameter is omitted, WAIT=Y is the default.

ECB

Specifies the address of an exp-type value (an address, reg, or L:addr type constant) which is the address of an ECB to be posted when this request has completed. The address of the specified ECB is placed in the NRB for this request. The ECB parameter can be used with WAIT=Y, N, or ADDRESS.

If WAIT=N or ADDRESS use an MVS WAIT macro to wait for this ECB, the ECB should be cleared before it is ever specified on a request. It should be cleared again after a wait on it has completed. If a user program may also post the same ECB, a check must be done to ensure that the ECB is not double posted.

SCONFIRM Entry Parameters

The following NRB fields are used by SCONFIRM on entry.

BUFA

address of outgoing data (move mode)

LEN

length of outgoing data

PROTA

buffer address for outgoing user ODATA

PROTL

length of the data contained in the buffer at the address specified in NRBPROTA

UBIT

data unused bit count

DMODE

datamode of data

SCONFIRM Results

The following NRB fields are updated when SCONFIRM completes.

STAT

success/failure code

LEN

set to zero

PROTL

set to zero

SREAD Assembler Request

The SREAD macro allows 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.

Optional Name	**Operation**	**Required Parameter**	**Optional Parameters**

`LABEL`	`SREAD`	`NRB=ADDRESS`	`,AREA=ADDRESS`
			`,LEN=DATALENGTH`
			`,OAREA=ADDRESS`
			`,OLEN=BUFLENGTH`
			`,ECB=ADDRESS`
			`,TIME=SECONDS`
			`,WAIT=Y`
			`,WAIT=N`
			`,WAIT=ADDRESS`

SREAD Parameters

LABEL

Specifies a standard Assembly language macro label.

NRB

This required expression specifies the address of the NRB data area to be passed to NetEx.

The following parameters may be specified in any order.

AREA

Specifies the starting address of the area to receive data from a CONFIRM, WRITE, DISCONNECT, or CLOSE request. For an SREAD to succeed, a valid buffer address must have been supplied through the NRB or the SREAD macro.

LEN

Specifies the length of the buffer used to hold data sent by the other application’s CONFIRM, WRITE, DISCONNECT, or CLOSE request.

OAREA

Specifies the starting address of the area to receive user ODATA sent as part of the message proper from a CONFIRM, WRITE, DISCONNECT, or CLOSE request.

OLEN

Specifies the length of the buffer used to hold user ODATA sent by the other application’s CONFIRM, WRITE, DISCONNECT, or CLOSE request.

ECB

This optional address is an exp-type value (an address, reg, or L:addr type constant), the address of a posted ECB when this request has completed. The address of the specified ECB is placed in the NRB for this request. The ECB parameter can be used with WAIT=Y, N, or ADDRESS.

If WAIT=N or ADDRESS use an MVS WAIT macro to wait for this ECB, the ECB should be cleared before it is ever specified on a request. It should be cleared again after a wait on it has completed. If a user program may also post the same ECB, a check must be done to ensure that the ECB is not double posted.

TIME
Specifies the length of time in seconds that the SREAD remains outstanding. A zero indicates no time limit. If not specified, the default is the last value used (the one remaining in the NRB). The programmer should take alternate path retry into consideration when specifying the timeout value.

WAIT
This is an optional parameter.

Y
This value indicates the user interface routine executes a WAIT (described in “Intertask Communication” on page 23).

N
This value indicates the program regains control as soon as NetEx has accepted the request.

If N is selected, the program may issue a SWAIT on this NRB or a series of NRBs at a later time, or do other work and occasionally check to see if the request has completed.

ADDRESS
This is an exp-type value (an address, reg, or L:addr type constant), the address of an asynchronous NRB post exit. For more information, refer to the following SWAIT request description.

If this parameter is omitted, WAIT=Y is the default.

SREAD Entry Parameters

The following NRB fields are used by SREAD on entry.

BUFA
address for incoming data (move mode)

BUFL
length of buffer to hold data

PROTA
buffer address for incoming user ODATA

PROTL
maximum length of the buffer at the address specified in NRBPROTA

ECB
address of the ECB to be posted (POST)

TIME
number of seconds offer outstanding

SREAD Results

The following NRB fields are updated when SREAD completes.

STAT
success/failure code

IND
contains connect indication

LEN
length of incoming data

PROTL
length of incoming user ODATA

UBIT
unused bit count of data

BLKO
maximum transmission data size (on read of confirm only)

BLKI
maximum reception data size (on read of confirm only)

SWRITE Assembler Request

The SWRITE macro allows an application 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 parameters existing in the specified NRB.

Optional Name	Operation	Required Parameters	Optional Parameters

`LABEL`	`SWRITE`	`NRB=ADDRESS`	`,AREA=ADDRESS`
			`,LEN=DATALENGTH`
			`,OAREA=ADDRESS`
			`,OLEN=BUFLENGTH`
			`,ECB=ADDRESS`
			`,MODE=MODE`
			`,WAIT=Y`
			`,WAIT=N`
			`,WAIT=ADDRESS`

SWRITE Parameters

LABEL

Specifies a standard Assembly language macro label.

NRB

This required expression specifies the address of the NRB data area to be passed to NetEx.

The following parameters may be specified in any order.

AREA

Specifies the address of a buffer of data to be sent to the corresponding application. The corresponding application receives this data with an SREAD macro and a Normal Data Indication (NRBNDTIN) in the NRBIND field.

LEN

Specifies the length of optional data (in addressable units or bytes for IBM) to be sent with the request. The length may range from zero addressable units (no data) up to the maximum declared by the BLKO parameter when the connection was established. Although the LEN and AREA parameters may be omitted, their omission causes the values specified during the previous request handled by the NRB to be used.

OAREA

Specifies the starting address of the area containing user ODATA to be sent as part of the message proper.

OLEN

Specifies the length of the data contained in the buffer specified by the OAREA parameter.

ECB

This optional parameters specifies the address is an exp-type value (an address, reg, or L:addr type constant) which is the address of an ECB to be posted when this request has completed. The address of the specified ECB is placed in the NRB for this request. The ECB parameter can be used with WAIT=Y, N, or ADDRESS.

If WAIT=N or ADDRESS use an MVS WAIT macro to wait for this ECB, the ECB should be cleared before it is ever specified on a request. It should be cleared again after a wait on it has completed. If a user program may also post the same ECB, a check must be done to ensure that the ECB is not double posted.

MODE

Specifies the assembly/disassembly modes to be used at both the transmitting and receiving adapters to deliver intelligible data to the receiving user. If omitted, the MODE specified during the previous use of the NRB is used. Refer to the datamode discussion "NRBDMODE" on page 46.

WAIT

This is an optional parameter.

Y

This value indicates the user interface routine executes a WAIT (described in "Intertask Communication" on page 23).

N

This value indicates the program regains control as soon as NetEx has accepted the request.

If N is selected, the program may issue an SWAIT on this NRB or a series of NRBs at a later time, or do other work and occasionally check to see if the request has completed.

ADDRESS

This is an exp-type value (an address, reg, or L:addr type constant), the address of an asynchronous NRB post exit. For more information, refer to the following SWAIT request description.

If this parameter is omitted, WAIT=Y is the default.

SWRITE Entry Parameters

The following NRB fields are used by SWRITE on entry.

BUFA

address of outgoing data

LEN

length of outgoing data

PROTA

buffer address for outgoing user ODATA

PROTL

length of the data contained in the buffer at the address specified in NRBPROTA

ECB

address of the ECB to be posted (POST)

UBIT

data unused bit count

DMODE

datamode of data

SWRITE Results

The following NRB fields are updated when SWRITE completes.

STAT
: success/failure code

LEN
: set to zero

PROTL
: set to zero

SWAIT Assembler Request

The SWAIT macro waits for the completion of one of several NetEx requests before processing continues in the application. The user receives control if any of the specified NRBs complete the operation in progress. If any of the specified NRBs has already completed when the SWAIT is issued, control returns to the application immediately.

The SWAIT request has three forms: an inline form, an execute form and a list form. The inline form generates a table of NRBs to wait upon in inline code, and calls the NetEx user interface to wait for one of them to complete. The list form generates a non-executable table of NRBs that can be used by the execute form SWAIT request.

If NetEx is not executing, a return code is given to the user in register 15.

Note: The `SWAIT....,MF=E` macro call uses register 2 along with the normal R0, R1, R14, R15 registers.

Optional Name	Operation	Required Parameters (Select One)	Optional Parameters	Required Parameters

LABEL	SWAIT	NRB=*NRB*,ECB=*ECB*
		NRB=(*NRB,NRB,...*),ECB=(*ECB,ECB,...*)
			,SSNM=*SSNM*
			,SSNM=NETX
			,MF=I	

		NRB=*ADDRESS*	,SSNM=*SSNM* ,SSNM=NETX	,MF=E

		NRB=*NRB*,ECB=*ECB*,MF=L
		NRB=(*NRB,NRB,...*),ECB=(*ECB,ECB,...*),MF=L		

		NRB=-1	,SSNM=*SSNM* ,SSNM=NETX ,MF=E	

		NRB=-2,ECB=*ECB*
		NRB=-2,ECB=(*ECB,ECB,...*)
			,SSNM=*SSNM*
			,SSNM=NETX
			,MF=I	

		NRB=-2,ECB=*ADDRESS*	,SSNM=*SSNM* ,SSNM=NETX	,MF=E

		NRB=-2,ECB=*ECB*
		NRB=-2,ECB=(*ECB,ECB,...*)
			,MF=L	

SWAIT Parameters

The following parameters were shown in the SWAIT request format. The parameters may be specified in any order.

LABEL

Specifies a standard Assembly language macro label.

Forms of SWAIT with Actual NRBs Specified

For MF=I (inline) macros (MF=I is the default):

NRB=NRB

Specifies the address of a single NRB that is to complete before control is returned to the application. The parameter must be an expression suitable for an A-type address constant. Either NRB=NRB or NRB=(NRB,NRB,...) must be specified.

NRB=(NRB,NRB,...)

These optional expressions are the addresses of the NRBs associated with two or more requests. Each parameter must be an expression suitable for an A-type address constant.

ECB=ECB

Specifies the address of a single ECB that is to be waited on along with the specified NRB(s). The parameter expression must be suitable for an A-type address constant.

ECB=(ECB, ECB,...)

These optional parameters are the addresses of ECBs to be waited for along with the specified NRB(s). Each parameter must be an expression suitable for an A-type address constant.

SSNM

Specifies the subsystem name of the NetEx to be used with this session. The default is NETX.

MF=I

This optional parameter tells NetEx to build a list from information specified by `NRB=` (used with inline code). This is the default if MF is not specified.

For MF=E (execute form) macros:

NRB=ADDRESS

This required parameter specifies the address of a list of NRBs, and possibly ECBs, that have been built by the user or a list form SWAIT macro instruction. This list may have one of two forms.

1. If the list includes only NRBs it can have the following form:

```
       address  DC   A(NRB1)
                DC   A(NRB2)
                         .
                         .
                         .
                DC   A(NRBn)
                DC   F’-1’
```

2. The following text is an example of the list format created by the list form of the SWAIT macro:

```
       address  DC   F’-2’
                DC   A(ECB1)
                DC   A(ECB2)
                         .
                         .
                         .
                DC   A(ECBn)
                DC   F’-2’
                DC   A(NRB1)
                         .
                         .
                         .
                DC   A(NRBn)
                DC   F’-1’
```

If there are no ECBs in this list, the first two words should be -2.

SSNM

Specifies the subsystem name of the NetEx to be used with this session. The default is NETX.

MF=E

This required parameter tells the NRB parameter to refers to a prebuilt list of NRB addresses.

For MF=L (list form) macros:

NRB=NRB

Specifies the address of a single NRB that is to complete before control is returned to the application. The parameter expression must be suitable for an A-type address constant. Either NRB=NRB or NRB=(NRB,NRB,...) must be specified.

NRB=(NRB,NRB,...)

These optional parameters are the addresses of the NRBs associated with two or more requests. Each parameter must be an expression suitable for an A-type address constant.

ECB=ECB

Specifies the address of a single ECB that is to be waited on along with the specified NRB(s). The parameter expression must be suitable for an A-type address constant.

ECB=(ECB, ECB,...)

These optional parameters are the addresses of ECBs to be waited for along with the specified NRB(s). Each parameter must be an expression suitable for an A-type address constant.

MF=L

This required parameter tells NetEx to build a list from information specified by `NRB=` to be used by the execute form SWAIT macro.

Forms of SWAIT Without NRBs Specified

These forms request a wait until any NRB for this task has completed.

NRB=-1

This required parameter asks to wait for completion of any NRB for this task.

MF=E

Specifies that this builds no inline lists and is re-enterant.

NRB=-2

This required parameter asks to wait for completion of any NRB for this task or any ECBs specified.

For MF=I (inline form of NRB=-2):

ECB=ECB

Specifies the address of a single ECB that is to be waited on along with the specified NRB(s) specified. The parameter expression must be suitable for an A-type address constant. Either ECB=ECB or ECB=(ECB, ECB,...) must be specified.

ECB=(ECB, ECB,...)

These optional parameters are the addresses of ECBs to be waited for along with the specified NRB(s). Each parameter must be an expression suitable for an A-type address constant.

SSNM

Specifies the subsystem name of the NetEx to be used with this session. The default is NETX.

MF=I

This optional parameter tells NetEx to build a list from information specified by the `ECB=` parameter (used with inline code). This is the default if MF is not specified.

For MF=E (execute form of NRB=-2):

ECB=ADDRESS

This required parameter specifies the address of a list of ECBs that has been built by the user or by the List form SWAIT macro instruction. The list's format is shown below:

```
      address  DC   F’-2’
```

```
                DC   A(ECB1)
                DC   A(ECB2)
                        .
                        .
                        .
                DC   A(ECBn)
                DC   F'-2'
                DC   F'-1'
```

SSNM

Specifies the subsystem name of the NetEx to be used with this session. The default is NETX.

MF=E

This required parameter tells the ECB parameter to refer to a prebuilt list of ECB addresses.

For MF=L (list form of NRB=-2):

ECB=ECB

Specifies the address of a single ECB that is to be waited on along with the specified NRB(s). The parameter expression must be suitable for an A-type address constant. Either ECB=ECB or ECB=(ECB, ECB,...) must be specified.

ECB=(ECB, ECB,...)

These optional parameters are the addresses of ECBs to be waited for along with the specified NRB(s). Each parameter must be an expression suitable for an A-type address constant.

MF=L

This required parameter tells NetEx to build a list from information specified by ECB= to be used by the execute form of the SWAIT with NRB=-2.

Note: Occasionally, an SWAIT-1 or -2 completes even though the user has acknowledged all NetEx requests. This may occur when NetEx marks event A completion, SWAIT-1 completes, control returns to the user, NetEx marks event B completion, user reviews all NRBs and finds that requests A and B have completed, user requests SWAIT-1 again, NetEx finds event B completion is marked, control returns to the user. To accommodate this sequence of events, the user program should be prepared to wait again.

Asynchronous Post Exit Processing

NetEx supports an asynchronous post exit for NRB posting which is dispatched when a NetEx request completes. This exit allows applications to get control when an event completes that was started with a WAIT=ADDRESS option (eliminating the need to issue the SWAIT primitive). The format of the parameter is described in each of the macro format descriptions.

This exit must use standard MVS type linkage conventions. An example is shown below:

```
    R1  = NRB address
    R13 = address of a 72-byte save area
    R14 = return address
    R15 = Exit start address
```

All registers must be saved and restored by the exit. This exit is used on a per NRB basis and is useful for applications that cannot afford to issue a wait while a NetEx request is being processed.

The ECB addressed in the NRB is posted when the exit is scheduled. This ECB may be the NRB ECB or it may be the ECB specified by the user program when the request was made to NetEx.

Posting (POST) occurs when NetEx is active or the request is valid (any non-user interface error). Otherwise, the ECB flag field and NRBSTAT field are marked complete upon completing the request without issuing the POST. The application should take this into consideration.

Note: The SWAIT facility issues SVCs; as a result, potential calling programs have the following limitations:

- May not execute in SRB mode.
- May not execute in physically disabled mode.
- May not hold any locks.
- May not be in cross-memory mode.

Violation of these rules result in a S0F8 ABEND.

SCLOSE Assembler Request

The SCLOSE macro allows a program to transmit data to another corresponding calling program and indicate that this is the last data to be sent for a session. The data must be received by a read request in the other program.

After a program issues an SCLOSE on a session, no other data may be written by that program on that session. If another program previously issued an SCLOSE, the data is written and the connection disconnects. If the other program has not issued an SCLOSE, it is still free to write data to the program session that issued the SCLOSE.

Before issuing the SCLOSE, a connection must be fully established.

Defaults for unspecified parameters are assumed to be the parameters existing in the NRB.

Optional Name	Operation	Required Parameter	Optional Parameters
LABEL	SCLOSE	NRB=*ADDRESS*	,AREA=*ADDRESS* ,LEN=*DATALENGTH* ,MODE=*MODE* ,OAREA=*ADDRESS* ,OLEN=*DATALENGTH* ,ECB=*ADDRESS* ,WAIT=Y ,WAIT=N ,WAIT=*ADDRESS*

SCLOSE Parameters

LABEL

Specifies standard Assembly language macro label.

NRB

This required expression specifies the address of the NRB data area to be passed to NetEx.

The following parameters may be specified in any order.

AREA

Specifies the address of a buffer of data to be sent to the corresponding application. The corresponding application receives this data with an SREAD macro and a Close Indication (NRBCLSIN) in the NRBIND field.

LEN

Specifies the length of optional data (in addressable units or bytes for IBM) to be sent with the request. The length may range from zero addressable units (no data) up to the maximum declared by the BLKO parameter when the connection was established. Although the LEN and AREA parameters may be omitted, their omission causes the values specified during the previous request handled by the NRB to be used.

MODE

Specifies the assembly/disassembly modes to be used at both the transmitting and receiving adapters to deliver intelligible data to the receiving user. If omitted, the MODE specified during the previous use of the NRB is used. Refer to the datamode discussion in "NRBDMODE" on page 46.

OAREA

Specifies the starting address of the buffer that contains data to be sent in the message proper.

OLEN

Specifies the length of the data contained in the buffer specified in OAREA.

ECB

Specifies the address is an exp-type value (an address, reg, or L:addr type constant) which is the address of an ECB to be posted when this request has completed. The address of the specified ECB is placed in the NRB for this request. The ECB parameter can be used with WAIT=Y, N, or ADDRESS.

If WAIT=N or ADDRESS use an MVS WAIT macro to wait for this ECB, the ECB should be cleared before it is ever specified on a request. It should be cleared again after a wait on it has completed. If a user program may also post the same ECB, a check must be done to ensure that the ECB is not double posted.

WAIT

This is an optional parameter.

Y

This value indicates the user interface routine executes a WAIT (described in “Intertask Communication” on page 23).

N

This value indicates the program regains control as soon as NetEx has accepted the request.

If N is selected, the program may issue a SWAIT on this NRB or a series of NRBs at a later time, or do other work and occasionally check to see if the request has completed.

ADDRESS

This is an exp-type value (an address, reg, or L:addr type constant), the address of an asynchronous NRB post exit. For more information, refer to the following SWAIT request description.

If this parameter is omitted, WAIT=Y is the default.

SCLOSE Entry Parameters

The following NRB fields are used by SCLOSE on entry.

BUFA

address of outgoing data

LEN

length of outgoing data

UBIT

data unused bit count

DMODE

datamode of data

PROTA

address of the data to be sent in the message proper

PROTL

length of the data to be sent in the message proper

ECB

address of the ECB to post

SCLOSE Results

The following NRB fields are updated when SCLOSE completes.

STAT
: success/failure code

LEN
: set to zero

PROTL
: set to zero

SDISCONN Assembler Request

The SDISCONN (disconnect) macro allows 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, the user must wait for confirmation of previous SREAD or SWRITE commands by the corresponding application before issuing the SDISCONN macro.

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

Optional Name	**Operation**	**Required Parameter**	**Optional Parameters**

`*LABEL*`	`SDISCONN`	`NRB=`*`ADDRESS`*	`,AREA=`*`ADDRESS`*
			`,LEN=`*`DATALENGTH`*
			`,MODE=`*`MODE`*
			`,OAREA=`*`ADDRESS`*
			`,OLEN=`*`DATALENGTH`*
			`,ECB=`*`ADDRESS`*
			`,WAIT=Y`
			`,WAIT=N`
			`,WAIT=`*`ADDRESS`*

SDISCONN Parameters

LABEL

Specifies a standard Assembly language macro label.

NRB

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

The following parameters may be specified in any order.

AREA

Specifies the address of a buffer of data to be sent to the corresponding application. The corresponding application receives this data with an SREAD macro and a disconnect indication in the NRBIND field.

LEN

Specifies the length of optional data (in addressable units or bytes for IBM) to be sent with the request. The length may range from zero up to the minimum of: the local host limit; the remote host limit; or the maximum allowed on the path. Although the LEN and AREA parameters may be omitted, their omission causes the values specified during the previous request handled by the NRB to be used.

MODE

Specifies the assembly/disassembly modes to be used at both the transmitting and receiving adapters to deliver intelligible data to the receiving user. If omitted, the MODE specified during the previous use of the NRB is used. Refer to the datamode in "NRBDMODE" on page 46.

OAREA

Specifies the starting address of the buffer that contains data to be sent in the message proper.

OLEN

Specifies the length of the data contained in the buffer specified in OAREA.

ECB

Specifies the address is an exp-type value (an address, reg, or L:addr type constant) which is the address of an ECB to be posted when this request has completed. The address of the specified ECB is placed in the NRB for this request. The ECB parameter can be used with WAIT=Y, N, or ADDRESS.

If WAIT=N or ADDRESS use an MVS WAIT macro to wait for this ECB, the ECB should be cleared before it is ever specified on a request. It should be cleared again after a wait on it has completed. If a user program may also post the same ECB, a check must be done to ensure that the ECB is not double posted.

WAIT

This is an optional parameter.

Y

Indicates the user interface routine executes a WAIT (described in “Intertask Communication” on page 23).

N

Indicates the program regains control as soon as NetEx has accepted the request.

If N is selected, the program may issue a SWAIT on this NRB or a series of NRBs at a later time, or do other work and occasionally check to see if the request has completed.

address

This is an exp-type value (an address, reg, or L:addr type constant), the address of an asynchronous NRB post exit. For more information, refer to the following SWAIT request description.

If this parameter is omitted, WAIT=Y is the default.

SDISCONN Entry Parameters

The following NRB fields are used by SDISCONN on entry.

BUFA

address of outgoing data

LEN

length of outgoing data

UBIT

data unused bit count

DMODE

datamode of data

ECB

address of the ECB to post

PROTA

address of the data to be sent in the message proper

PROTL

length of the data to be sent in the message proper

SDISCONN Results

The following NRB fields are updated when SDISCONN completes.

STAT
    success/failure code

LEN
    set to zero

PROTL
    set to zero

C High Level Programming Interface

NETEX includes a library of subroutines that are designed to be called by C high level language programs. Also included are global data declarations and external reference declarations that may be included at compile time. This file is called *netex.h*. When the user makes a call to the user interface, the appropriate information is supplied in parameter format to pass to NETEX.

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 C 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.

soffr	offer services
sconn	connect to an offered program
sconf	confirm acceptance of connect
sread	read incoming request or data
swrit	write data
sclos	write last data
swait	wait for previous request(s) to complete
sdisc	disconnect (immediate)

These calls are described in “Session Layer Requests”.

C NETEX Request Blocks

The C user creates an NRB by declaring a data structure of 21 fields using the NETEX supplied data structure template NRB. Various fields of this record will hold the information to be transferred to NETEX, and others will contain the information that is returned by NETEX. NRB variables should be declared to be of that type. Before these NRB records are used for any NETEX request, the caller is advised to zero all the elements of the record. This will allow defaults for fields not explicitly used by the caller to take effect.

The NETEX C 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 field of the NRB record. When the request completes, the C program directly accesses the desired fields of the NRB record to determine whether the operation completed properly. If NRB is declared as a 21 field record, the fields of the record will be defined using the following:

Field	**Type**	**Function**
NRBSTAT	long	Status returned from NETEX
NRBIND	long	Data type indication from OFFER/READ
NRBLEN	long	Length of data received in words
NRBUBIT	long	Unusedbitcount
NRBREQ	long	Requestcode
NRBNREF	char*	NETEX Reference number (N-ref)forthissession
NRBBUFA	long	User'sPdata buffer address
NRBBUFL	long	Length of the buffer
NRBDMODE	long	Datamode for WRITE request
NRBTIME	long	Timeout for a read type request (in seconds)
NRBCLASS	long	Class of service
NRBMAXRT	long	Maximum rate of data transmission
NRBBLKI	long	Blocksize to use for input
NRBBLKO	long	Block size to use for output
NRBPROTA	char*	User's Odata buffer address
NRBPROTL	long	Length of Odata buffer
NRBRESV1	long	Reserved
NRBRESV2	long	Reserved
NRBOFFER	char:8	Offer Name (Source)
NRBHOST	char:8	Logical name of destination host

NRBRESV3	long	Reserved
NRBRESV4	long	Reserved
NRBUSER	long:1	Free for user to assign (nrbuser=nrbresv4)
NRBOSDEP	long:16	Reserved for system dependent information

Figure 12. 'C' NETEX Request Block (NRB)

SOFFR C Function

The SOFFR (offer) and SOFFRW (offer wait) functions make the services, provided by the calling NETEX application program, available to programs running on other hosts. The SOFFR is actually a specialized form of read request. The SOFFR reads any data associated with the SCONN.

Before issuing an SOFFR call, the user must provide an NRB (described in “NetEx Request Block” on page 41) to be used by the user interface.

SOFFR Function Format

The SOFFR function has the following format:

Function (Select One)	**Required Parameters**
`soffr` `soffrw`	`(nrb,buffer,length,timeout,pname[,ssnm])`

SOFFR Parameters

The following parameters were shown in the SOFFR function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters’ positions.

soffr
soffrw

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

nrb

(struct nrb *) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char *) This required parameter is the extended address of the buffer data area to receive data sent by the corresponding SCONNECT request.

length

(long:VALUE) This required parameter is the length of the buffer (in addressable units) 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 field will contain the number of bytes of information actually sent to the offering application.

timeout

(long:VALUE) This required parameter is 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 *timeout* is specified as zero, the OFFER will remain outstanding indefinitely.

pname

char*:REF:) This required 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 must be provided as a string in the CALL statement padded with blanks to eight characters in length.

ssnm

This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

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
NRBPROTA	Address for incoming Odata
NRBPROTL	Length of buffer to hold Odata
NRBTIME	Number of seconds offer outstanding
NRBBLKO	Maximum transmission size acceptable
NRBBLKI	Maximum reception size acceptable
NRBMXRAT	Limit on transmission speed
NRBOFFER	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
NRBPROTL	Length of Odata received
NRBNREF	S-ref assigned this connection
NRBBLKO	Maximum transmission Pdata size
NRBBLKI	Maximum reception Pdata size
NRBMXRAT	Maximum transmission speed of path
NRBHOST	Name of host where S-conn originated

SCONN C Function

The SCONN (connect) function provides a means for a program to request a session with a program that has issued an SOFFR. The SCONN is a specialized form of a write request. The SCONN initiates the session and may optionally write data to the offerer at the same time.

Before issuing the SCONN, an NRB must be provided for use by the user interface.

SCONN Function Format

The SCONN function has the following format:

Function (Select One)	**Required Parameters**
`sconn` `sconnw`	`(nrb,buffer,length,datamd,pname,hname[,ssnm])`

SCONN Parameters

The following parameters were shown in the SCONN function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters' positions.

sconn
sconnw

This is the verb for this function. SCONNW specifies that the calling program must wait for the call to complete before processing is resumed.

nrb

(struct nrb*) This required parameter is the pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char*) This required parameter is the extended address of the buffer data area that holds the user data to be sent to the corresponding application.

length

(INT(32):VALUE) This required parameter is the length of the data (in addressable units) to be sent to the corresponding program. This value is 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.

datamd

(long:VALUE) This required parameter is the datamode to be used to send the connect data to the corresponding application. Refer to NRBDMODE in "NetEx Request Block" for a discussion of the datamode parameter.

pname

(char*REF:) This required parameter is the alphabetic name of the process offered (SOFFR) by the corresponding calling program. The name offered is determined by the other calling program. This name must be provided as a string in the call invocation, padded with blanks to eight characters in length.

hname

(char*REF:) This required parameter is 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 must be provided as a string in the call invocation, padded with blanks to eight characters in length.

ssnm

This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

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
NRBPROTA	Address of outgoing Odata
NRBPROTL	Length of outgoing Odata
NRBBLKO	Maximum transmission size acceptable
NRBBLKI	Maximum reception size acceptable
NRBMXRAT	Limit on transmission speed
NRBHOST	Alphanumeric “host” name
NRBOFFER	Alphanumeric “application” name

SCONN Results

The following NRB fields are updated when SCONN completes.

NRBSTAT	Success/failure code
NRBNREF	S-ref (Session ID) assigned
NRBBLKO	Maximum transmission Pdata size
NRBBLKI	Maximum reception Pdata size
NRBMXRAT	Maximum transmission speed of path

SCONF C Function

The SCONF (confirm) function 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.

The SCONF is a specialized form of write request. Data may be optionally written during the confirm process with this command.

Before issuing the SCONF function, 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.

SCONF Function Format

The SCONF function has the following format:

Function (Select One)	**Required Parameters**
`sconf` `sconfw`	`(nrb,buffer,length,datamd[,ssnm])`

SCONF Parameters

The following parameters were shown in the SCONF function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters' positions.

sconf
sconfw

This is the verb for this function. SCONFW specifies that the calling program must wait for the call to complete before processing is resumed.

nrb

(struct nrb*) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char*) This required parameter is the extended address of the buffer data area that holds the user data to be sent to the corresponding application.

length

(long:VALUE) This required parameter is the length of the data (in addressable units) to be sent to the corresponding program. This value is 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.

datamd

(long:VALUE) This required parameter is the datamode to be used to send the connect data to the corresponding application. Refer to NRBDMODE in "NetEx Request Block" for a discussion of the datamode parameter.

ssnm

This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

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 C Function

The SREAD subroutine 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.

Important: Keep an SREAD request outstanding to receive incoming data. NETEX will automatically terminate a connection if a request is waiting to be read for too long. This read-timeout value is set at installation time.

Defaults for unspecified parameters are assumed to be the parameters existing in the NRB. After BUFFER and LENGTH are agreed on during the connection process, these parameters can be omitted.

SREAD Function Format

The SREAD function has the following format:

Function (Select One)	**Required Parameters**
`sread` `sreadw`	`(nrb,buffer,length,timeout[,ssnm])`

SREAD Parameters

The following parameters were shown in the SREAD function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters' positions.

sread
sreadw

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

nrb

(struct nrb*) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char *) This required parameter is the extended address of the buffer data area to receive the data sent by the corresponding application's SWRITE or SCONF request.

length

(long:VALUE) This required parameter is the length of the buffer (in addressable units) 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. Programs that wish to work with the Unused Bit Count on input should examine the NRBUBIT field.

timeout

(long:VALUE) This required parameter is 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 *timeout* is specified as zero, the READ will remain outstanding indefinitely.

ssnm

This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

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
NRBPROTA	Address for incoming Odata
NRBPROTL	Length of buffer to hold Odata
NRBTIME	Number of seconds offer outstanding

SREAD Results

The following NRB fields are updated when SREAD completes.

NRBSTAT	Success/failure code
NRBIND	Contains Connect Indication
NRBLEN	Length of incoming Pdata
NRBUBIT	Unused bit count of Pdata
NRBPROTL	Length of Odata received
NRBBLKO	Maximum transmission Pdata size (On Read of Confirm only)
NRBBLKI	Maximum reception Pdata size (On Read of Confirm only)

SWRIT C Function

The SWRIT (write) function provides a means for a program to transmit data to another calling program.

Before an SWRIT can be issued, a connection must be established.

SWRIT Function Format

The SWRIT function has the following format:

Function (Select One)	**Required Parameters**
`swrit` `swritw`	`(nrb,buffer,length,datamd[,ssnm])`

SWRIT Parameters

The following parameters were shown in the SWRIT function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters' positions.

swrit
swritw

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

nrb

(struct nrb *) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char *) This required parameter is the extended address of the buffer data area that holds the user data to be sent to the corresponding application.

length

(long:VALUE) This required parameter is the length of the data (in addressable units) to be sent to the corresponding application. The *length* may be set to zero.

datamd

(long:VALUE) This required parameter is the datamode to be used to send the connect data to the corresponding application. Refer to NRBDMODE in "NetEx Request Block" for a discussion of the datamode parameter.

ssnm

This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

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 C Function

The SCLOS (close) function provides a means for a program to transmit data to another corresponding calling program and indicates that this is the last data to be sent. The data must be received by a read request in the other program.

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

Before issuing the SCLOS, a connection must be fully established.

SCLOS Function Format

The SCLOS function has the following format:

Function (Select One)	**Required Parameters**
`sclos` `sclosw`	`(nrb,buffer,length,datamd[,ssnm])`

SCLOS Parameters

The following parameters were shown in the SCLOS function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters' positions.

sclos
sclosw

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

nrb

(struct nrb*) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char*) This required parameter is the extended address of the pointer to the buffer data area that holds the user data to be sent to the corresponding application.

length

(long:VALUE) This required parameter is the length of the data (in addressable units) to be sent to the corresponding application. The *length* may be set to zero.

datamd

(long:VALUE) This required parameter is the datamode to be used to send the connect data to the corresponding application. Refer to NRBDMODE in "NetEx Request Block" for a discussion of the datamode parameter.

ssnm

This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

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
NRBBUFA	Contains zero

SWAIT C Function

The SWAIT function provides a means to wait for NETEX completions that were issued "nowaited." The action taken by the subroutine will be different based on the value in the parameter NRBNUM.

If NRBNUM is greater than zero (swait specific) the subroutine will return control to the caller when any request identified in the NRB list has completed.

If NRBNUM is equal to zero, the subroutine will attempt to post all completed requests and return control to the caller immediately. If a previous call to SWAIT using NRBNUM equal to minus two (-2) has been issued but has not completed, the current SWAIT call will have no effect.

If NRBNUM is equal to minus one (-1), the subroutine will attempt to post all completed requests. If there were no completions at the time of the call, the subroutine will wait until a request has completed before returning control to the caller.

If NRBNUM is equal to minus two (-2), the action is system dependent.

SWAIT C Function Format

The SWAIT function has the following format:

Function	Parameters
`swait`	`(nrbnum,nrb,nrb...,nrb10[,ssnm])`
`swait`	`(-2L[,ssnm])`

SWAIT Parameters

The following parameters were shown in the SWAIT function format. The parameters must be specified in the order presented. If parameters are omitted, then commas must be used to preserve subsequent parameters' positions.

swait

This is the verb for this function.

nrbnum

(long:VALUE) This required parameter is the number of NRBs in the nrb or one of the values described above.

nrb

(struct nrb*) This required parameter is an address pointer to one or more NRBs (the number of NRBs specified in nrbnum) associated with the request to be waited for (maximum of 10). An *nrb* is required for each NRB specified in *nrbnum*.

ssnm

This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

The SWAIT call provides the means to wait for the completion of requests that have not been previously waited for. Control will be returned to the SWAIT caller when 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. By waiting on 0 NRBs, the NETEX subroutine library will take control and update all NRBs, after which it will return control to the user.

After control is returned, the calling C program must determine which of the NRBs in the list has completed. This can be done by examining the NRBSTAT field of each of the NRBs.

SDISC C Function

The SDISC (disconnect) function 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, the operator must wait for confirmation of previous SREAD or SWRIT calls before issuing the SDISC.

When an SDISC is issued, an outstanding SREAD in the other program will terminate with an error in NRBSTAT.

NETEX does not ensure that data written with an SDISC macro will actually be received by the other program.

SDISC Function Format

The SDISC function has the following format:

Function (Select One)	**Required Parameters**
`sdisc` `sdiscw`	`(nrb,buffer,length,datamd[,ssnm])`

SDISC Parameters

The following parameters were shown in the SDISC function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters' positions.

sdisc
sdiscw

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

nrb

(struct nrb*) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char*) This required parameter is the extended address of the buffer data area that holds the user data to be sent to the corresponding application.

Note: In the single case of SDISC, delivery of DISCONNECT data is NOT reliable, although the actual disconnection will always occur.

length

(long:VALUE) This required parameter is the length of the data (in addressable units) to be sent to the corresponding program. This value is 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.

datamd

(long:VALUE) This required parameter is the datamode to be used to send the disconnect data to the corresponding application. Refer to NRBDMODE in "NetEx Request Block" for a discussion of the datamode parameter.

On 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.

ssnm

This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

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 Programming Interface

General

This section describes the following topics:

- FORTRAN high level interface
- FORTRAN NetEx Request Blocks
- SOFFR FORTRAN Call
- SCONN FORTRAN Call
- SCONF FORTRAN Call
- SREAD FORTRAN Call
- SWRIT FORTRAN Call
- SWAIT FORTRAN Call
- SCLOS FORTRAN Call
- SDISC FORTRAN Call

FORTRAN high level interface

NetEx includes a library of subroutines that can be called by FORTRAN high level language programs. When the user interface is called, the user supplies the appropriate information in parameter format to pass to NetEx.

Two components 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):

- SOFFR
- SCONN
- SCONF
- SREAD
- SWRIT
- SWAIT
- SCLOS
- SDISC

These calls are described in “Intertask Communication” on page 23. 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 “H210IPZ NetEx/IP Overview” on page 15.

FORTRAN NetEx Request Blocks

The FORTRAN user builds an NRB by declaring it to be an array of 40 INTEGER*4 elements. Various members of this array hold the information transferred to NetEx, and others contain the information returned by NetEx. If more than one NRB services an application, then several NRB arrays must be declared. Before these NRB arrays are used for any NetEx request, Network Executive Software advises to zero all the elements of the array. This allows defaults for fields not explicitly used by the caller to take effect. Thus a sample declaration might be:

```
            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, is passed as parameters to the subroutine. Exotic parameters to be passed to NetEx, such as maximum input block size, is 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 is as shown in Table 5.

Table 5. FORTRAN NetEx Request Block

Element	Type	Name	Function
NRB(1)	INTEGER*4	NRBSTAT	Status returned from NetEx
NRB(2)	INTEGER*4	NRBIND	Read indication type
	1	NRBCONIN	Connect Indication
	2	NRBCNFIN	Confirm Indication
	3	NRBNDTIN	Normal Data Indication
	4	NRBEDTIN	Expedited Data Ind. (reserved)
	5	NRBCLSIN	Close Indication
	6	NRBDISIN	Disconnect Indication
	7	NRBSTSIN	Status indication (reserved)
NRB(3)	INTEGER*4	NRBLEN	Length of data received in addressable units
NRB(4)	INTEGER*4	NRBUBIT	Unused bit count
NRB(5)	INTEGER*4	NRBREQF	Request code
NRB(6)	INTEGER*4	NRBNREF	NREF for this session
NRB(7)	INTEGER*4	NRBBUFA	User’s data buffer address
NRB(8)	INTEGER*4	NRBBUFL	Length of the buffer
NRB(9)	INTEGER*4	NRBDMODE	Assembly/disassembly mode
NRB(10)	INTEGER*4	NRBTIME	Timeout for a read type request
NRB(11)	INTEGER*4	NRBCLASS	Type of protocol
NRB(12)	INTEGER*4	NRBMAXRT	Maximum rate of transmission
NRB(13)	INTEGER*4	NRBBLKI	Block size to use for input
NRB(14)	INTEGER*4	NRBBLKO	Block size to use for output
NRB(15)	INTEGER*4	NRBPROTA	User’s ODATA buffer address
NRB(16)	INTEGER*4	NRBPROTL	Length of the buffer
NRB(17)	INTEGER*4	NRBRESV1	Reserved

Table 5. FORTRAN NetEx Request Block			
Element	**Type**	**Name**	**Function**
NRB(18)	INTEGER*4	NRBRESV2	Reserved
NRB(19)	CHARACTER*8	NRBPNAME	Logical Process name (Source)
NRB(20)	CHARACTER*8	NRBHNAME	Logical name of destination host
NRB(21)	INTEGER*4	NRBRESV3	Reserved
NRB(22)	INTEGER*4	NRBRESV4	Reserved
NRB(25) - NRB(40)	Reserved: system-dependent information		

SOFFR FORTRAN Call

The SOFFR (offer) subroutine allows 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" on page 41) to be used by the user interface. NetEx must be active in the system.

SOFFRW tells the calling program to wait for the call to complete before it resumes processing.

Call	**Operation (Select One)**	**Parameters (Select One Group)**

`CALL`	`SOFFR`
	`SOFFRW`
		`(NRB,BUFFER,LENGTH,TIMEOU,PNAME)`
		`(NRB,BUFFER,LENGTH,TIMEOU,PNAME,SSNM)`
		`(NRB,BUFFER,LENGTH,TIMEOU,PNAME,``NETX`*`)`*

SOFFR Parameters

The SOFFR call parameters are described below. The parameters must be specified in the order presented.

CALL
This is the standard high level call instruction.

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

BUFFER
This required parameter specifies the start of an array to receive data sent by the corresponding SCONNECT request.

LENGTH
This required parameter is the length of the buffer (in addressable units or bytes for IBM) that holds 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 contains the number of addressable units of information actually sent to the offering application. The data type of length should be INTEGER*4.

TIMEOU
This required parameter is the number of seconds that the OFFER request should remain outstanding. If no application connects during this interval, then the OFFER ends abnormally. If timeou is specified as zero, the OFFER remains outstanding indefinitely.

PNAME
This required parameter specifies 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 long (left justified blank filled).

SSNM
This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

SOFFR Entry Parameters

The following NRB fields are used by SOFFR on entry.

BUFA

address for incoming data

BUFL
length of buffer to hold data

PROTA
address for incoming ODATA

PROTL
length of buffer to hold ODATA

TIME
number of seconds offer outstanding

CLASS
class of service requested

BLKO
maximum transmission size acceptable

BLKI
maximum reception size acceptable

MAXRT
limit on transmission speed

PNAME
application name to offer

SSNM
subsystem name used to interface to NetEx

SOFFR Results

The following NRB fields are updated when SOFFR completes.

STAT
success/failure code

IND
contains connect indication

LEN
length of incoming data

PROTL
length of incoming ODATA

UBIT
unused bit count of data

NREF
SREF assigned this connection

CLASS
class of service assigned

BLKO
maximum transmission data size

BLKI

maximum reception data size

MAXRT
maximum transmission speed of path

HNAME
name of host where SCONN originated

HNAME
name of application where SCONN originated

SSNM
subsystem name of the local NetEx

SCONN FORTRAN Call

The SCONN (connect) call allows 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.

SCONNW tells the calling program to wait for the call to complete before it resumes processing.

Call	Operation (Select One)	Parameters (Select One Group)
`CALL`	`SCONN` `SCONNW`	*(NRB,BUFFER,LENGTH,DATAMD,PNAME,HNAME)* *(NRB,BUFFER,LENGTH,DATAMD,PNAME,HNAME,SSNM)* *(NRB,BUFFER,LENGTH,DATAMD,PNAME,HNAME,*<u>NETX</u>*)*

SCONN Parameters

The SCONN call format parameters are described below. The parameters must be specified in the order presented.

CALL
This is the standard high level call instruction.

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

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

LENGTH
This required parameter specifies the length of the data (in addressable units or bytes for IBM) 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" on page 46 for a discussion of the datamode parameter.

PNAME
This required parameter specifies the alphabetic name of the process offered (SOFFR) 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 (left justified blank filled).

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 (left justified blank filled).

SSNM
This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

SCONN Entry Parameters

The following NRB fields are used by SCONN on entry.

BUFA
address of outgoing data

LEN
length of outgoing data

PROTA
address of outgoing ODATA

PROTL
length of outgoing ODATA

UBIT
data unused bit count

DMODE
datamode of data

CLASS
class of service requested

BLKO
maximum transmission size acceptable

BLKI
maximum reception size acceptable

MAXRT
limit on transmission speed

HNAME
alphanumeric host name

PNAME
alphanumeric application name

SSNM
subsystem name used to interface to NetEx

SCONN Results

The following NRB fields are updated when SCONN completes.

STAT
success/failure code

LEN
set to zero

PROTL
set to zero

NREF	SREF (Session ID) assigned
CLASS	class of service assigned
BLKO	maximum transmission data size
BLKI	maximum reception data size
MAXRT	maximum transmission speed of path
SSNM	subsystem name of the local NetEx

SCONF FORTRAN Call

The SCONF (confirm) call allows 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.

SCONFW tells the calling program to wait for the call to complete before it resumes processing.

Call	**Operation (Select One)**	**Parameters**
`CALL`	`SCONF` `SCONFW`	*`(NRB,BUFFER,LENGTH,DATAMD)`*

SCONF Parameters

The SCONF call parameters are described below. The parameters must be specified in the order presented.

CALL
This is the standard high level call instruction.

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

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

LENGTH
This required parameter specifies the length of the data (in addressable units or bytes for IBM) 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 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" on page 46 for a discussion of the datamode parameter.

SCONF Entry Parameters

The following NRB fields are used by SCONF on entry.

BUFA
address of outgoing data (move mode)

LEN
length of outgoing data

PROTA
address of outgoing ODATA (move mode)

PROTL
length of outgoing ODATA

UBIT
data unused bit count

DMODE
: datamode of data

SCONF Results

The following NRB fields are updated when SCONF completes.

STAT
: success/failure code

LEN
: set to zero

PROTL
: set to zero

SREAD FORTRAN Call

The SREAD macro allows 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.

SREADW tells the calling program to wait for the call to complete before it resumes processing.

Call	Operation (Select One)	Parameters
`CALL`	`SREAD` `SREADW`	`(`*`NRB,BUFFER,LENGTH,TIMEOU`*`)`

SREAD Parameters

The SREAD call parameters are described below. The parameters must be specified in the order presented.

CALL
This is the standard high level call instruction.

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

BUFFER
This required parameter specifies the start of an array 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 or bytes for IBM) 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 is 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 required 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 ends abnormally. If TIMEOU is specified as zero, the READ remains outstanding indefinitely. The programmer should take alternate path retry into consideration when specifying the timeout value. Refer to "Alternate Path Retry" in the *H210IPZ NetEx/IP for IBM z/OS Installation Reference Manual.*

SREAD Entry Parameters

The following NRB fields are used by SREAD on entry.

BUFA
address for incoming data (move mode)

BUFL
length of buffer to hold data

PROTA
address for incoming ODATA (move mode)

PROTL

length of buffer to hold ODATA

TIME
number of seconds offer outstanding

SREAD Results

The following NRB fields are updated when SREAD completes.

STAT
success/failure code

IND
contains connect indication

LEN
length of incoming data

PROTL
length of incoming ODATA

UBIT
unused bit count of data

BLKO
maximum transmission data size (on read of confirm only)

BLKI
maximum reception data size (on read of confirm only)

SWRIT FORTRAN Call

The SWRIT (write) call allows an application to transmit data to another calling program.

Before an SWRIT can be issued, a connection must be established.

SWRITW tells the calling program to wait for the call to complete before it resumes processing.

Call	Operation (Select One)	Parameters
`CALL`	`SWRIT` `SWRITW`	(*NRB,BUFFER,LENGTH,DATAMD*)

SWRIT Parameters

The SWRIT call parameters are described below. The parameters must be specified in the order presented.

CALL
This is standard high level call instruction.

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

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

LENGTH
This required parameter specifies the length of the data (in addressable units or bytes for IBM) 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" on page 46 for a discussion of the datamode parameter.

SWRIT Entry Parameters

The following NRB fields are used by SWRIT on entry.

BUFA
address of outgoing data

LEN
length of outgoing data

PROTA
address of outgoing ODATA

PROTL
length of outgoing ODATA

UBIT
data unused bit count

DMODE

datamode of data

SWRIT Results

The following NRB fields are updated when SWRIT completes.

STAT
: success/failure code

LEN
: set to zero

PROTL
: set to zero

SWAIT FORTRAN Call

The SWAIT call waits for the completion of one of several NetEx requests before processing continues in the application. Control returns to the SWAIT caller if any of the specified NRBs complete the operation in progress. If any one of the NRBs has completed before the SWAIT call, control returns to the application immediately. Up to 100 NRBs may be specified.

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

Call	**Operation**	**Optional Parameters (Select One Group)**

`CALL`	`SWAIT`	*`NRBNUM,NRB,NRB...`*
		`-1`
		`-1,` *`SSNM`*
		`-1, NETX`

SWAIT Parameters

The SWAIT call parameters are described below. The parameters must be specified in the order presented.

CALL
This is the standard high level call instruction.

NRBNUM
This parameter specifies the number of NRBs to wait for before processing continues. Control returns after any one of the specified NRBs complete; the maximum is 100.

NRB
This required parameter specifies the address of one or more NRBs (the number of NRBs specified in nrbnum) associated with the request.

-1
This parameter specifies that the SWAIT caller wishes to wait for any NetEx request for this caller. No NRBs are supplied on this request.

SSNM
This optional parameter is the subsystem (interface) name used to access NetEx. The default is NETX.

NOTE: There may be some cases in which an SWAIT-1 completes even though no new NetEx request has been completed that the user has not already acknowledged. The cause of this is as follows: NetEx marks event A completion, SWAIT-1 completes, control returns to the user, NetEx marks event B completion, user reviews all NRBs and finds that requests A and B have completed, user requests SWAIT-1 again, NetEx finds event B completion is marked, control returns to the user. To accommodate this sequence of events, the user program should be prepared to wait again.

SCLOS FORTRAN Call

The SCLOS (close) call allows 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.

SCLOSW tells the calling program to wait for the call to complete before it resumes processing.

Call	**Operation (Select One)**	**Parameters**
`CALL`	`SCLOS` `SCLOSW`	`(`*`NRB,BUFFER,LENGTH,DATAMD`*`)`

SCLOS Parameters

The SCLOS call parameters are described below. The parameters must be specified in the order presented.

CALL
This is the standard high level call instruction.

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

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

LENGTH
This required parameter specifies the length of the data (in addressable units, this is bytes for IBM) 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" on page 46 for a discussion of the datamode parameter.

When the SCLOS completes, the results are found in NRB(1) (NRBSTAT). This field informs the caller if the transfer succeeded and provides an error reason code if it did not.

SCLOS Entry Parameters

The following NRB fields are used by SCLOS on entry.

BUFA
address of outgoing data

LEN
length of outgoing data

PROTA
address of outgoing ODATA

PROTL
: length of outgoing ODATA

UBIT
: data unused bit count

DMODE
: datamode of data

SCLOS Results

The following NRB fields are updated when SCLOS completes.

STAT
: success/failure code

LEN
: set to zero

PROTL
: set to zero

SDISC FORTRAN Call

The SDISC (disconnect) call allows 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, the user must wait for confirmation of previous SREAD or SWRIT calls before issuing the SDISC.

Before an SDISC can be issued, the user must provide a NRB with an NRBNREF relating to this session.

SDISCW tells the calling program to wait for the call to complete before it resumes processing.

Call	Operation (Select One)	Parameters
`CALL`	`SDISC` `SDISCW`	(*NRB,BUFFER,LENGTH,DATAMD*)

SDISC Parameters

The SDISC call parameters are described below. The parameters must be specified in the order presented.

CALL
: This is the standard high level call instruction.

NRB
: This required parameter specifies the address of the NRB data area to be passed to NetEx.

BUFFER
: This required parameter specifies the start of an array that holds the user data to be sent to the corresponding application. Note that in the single case of SDISC, delivery of DISCONNECT data is not reliable, although the actual disconnection always occurs.

LENGTH
: This required parameter specifies the length of the data (in addressable units or bytes for IBM) 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 required parameter specifies the datamode to be used to send the disconnect data to the corresponding application. The data type of DATAMD should be INTEGER. Refer to "NRBDMODE" on page 46 for a discussion of the datamode parameter.

Upon completion of the SDISC, the connection no longer exists -- new requests against that connection are 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.

BUFA
: address of outgoing data

LEN
: length of outgoing data

PROTA
: address of outgoing data

PROTL

length of outgoing data

UBIT
data unused bit count

DMODE
datamode of data

SDISC Results

The following NRB fields are updated when SDISC completes.

STAT
success/failure code

LEN
set to zero

PROTL
set to zero

PASCAL Programming Interface

General

This section describes the following topics:

- PASCAL High Level Interface
- PASCAL NetEx Request Blocks
- SOFFR PASCAL Call
- SCONN PASCAL Call
- SCONF PASCAL Call
- SREAD PASCAL Call
- SWRIT PASCAL Call
- SWAIT PASCAL Call
- SCLOS PASCAL Call
- SDISC PASCAL Call

PASCAL High Level Interface

NetEx includes a library of subroutines that are designed to be called by PASCAL high level language programs. Also included is a library of copyfiles that may be included (%INCLUDE) into a PASCAL program and inserted at compile time. When making a call to the user interface, user supplies the appropriate information, in parameter format, to pass to NetEx.

Two components are used to establish working communications through NetEx: one or more NetEx Request Blocks (NRBs) that must be supplied by the PASCAL 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):

- SOFFR
- SCONN
- SCONF
- SREAD
- SWRIT
- SWAIT
- SCLOS
- SDISC

These calls are described in “Intertask Communication” on page 23. The formats of the calls are presented using the conventions stated in “H210IPZ NetEx/IP Overview” on page 15.

PASCAL NetEx Request Blocks

The PASCAL user builds an NRB by declaring it to be a record of 21 fields. Some of the fields for this record contain information that is transferred to NetEx, and other fields contain information that is returned by NetEx. If more than one NRB services an application, one NRB type should be defined and several NRB variables should be declared to be of that type. Before these NRB records are used for any NetEx request, Network Executive Software advises to zero all the elements of the record. This allows defaults for fields not explicitly used by the caller to take effect.

The NetEx PASCAL subroutines have the philosophy that arguments commonly passed to NetEx, such as a data buffer address, are passed as parameters to the subroutine. Exotic parameters to be passed to NetEx, such as maximum input block size, are supplied by storing the desired value in the proper field of the NRB record. When the request completes, the PASCAL program directly accesses the desired fields of the NRB record to determine if the operation completed properly. If the NRB is declared as a 21 field record, the fields of the record are defined using the following types:

```
CONST
   NullIND      = 0;
   ConnIND      = 1;
   ConfIND      = 2;
   NormalIND    = 3;
   ExpeditedIND = 4;
   CloseIND     = 5;
   DisconIND    = 6;
   StatusIND    = 7;
TYPE
   NRBname = packed array(.1..8.);
```

The fields and corresponding types are as shown in Table 6.

Table 6. PASCAL NetEx Request Block

Field	**Type**	**Function**
Status	INTEGER	Status returned from NetEx
Indication	NullIND..Status	Read indication: may be 1 of 7 types
	ConnIND	Connect Indication
	ConfIND	Confirm Indication
	NormalIND	Normal Data Indication
	ExpeditedIND	Expedited Data Indication (reserved)
	CloseIND	Close Indication
	DisconIND	Disconnect Indication
	StatusIND	Status indication (reserved)
Length	INTEGER	Length of data received in words
UnUsedBits	INTEGER	Unused bit count
Request	INTEGER	Request code
Nref	INTEGER	NREF for this session
BufAddr	INTEGER	User’s data buffer address
BufLen	INTEGER	Length of the buffer
DataMode	INTEGER	Assembly/disassembly mode
TimeOut	INTEGER	Timeout for a read type request
Class	INTEGER	Type of protocol
MaxRate	INTEGER	Maximum rate of transmission
BlockIn	INTEGER	Block size to use for input
BlockOut	INTEGER	Block size to use for output
OdatAddr	INTEGER	Msg proper - driver request only
OdatLen	INTEGER	Length of msg proper-driver request
Reserved	(.1..2.) of INTEGER	Reserved
ApplicName	NRBname	Logical Process name (Source)
HostName	NRBname	Logical name of destination host
Resrv3	(.3....) of INTEGER	Reserved
Resrv4	(.4....) of INTEGER	Reserved
OSdep	(.1..16.) of INTEGER	Reserved for system-dependent information

SOFFR PASCAL Call

The SOFFR (offer) subroutine allows 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" on page 41) to be used by the user interface. NetEx must be active in the system.

SOFFRW tells the calling program to wait for the call to complete before it resumes processing.

Operation (Select One)	**Required Parameters (Select One Group)**

`SOFFR`
`SOFFRW`
	`(`*`NRB,BUFFER,LENGTH,TIMEOU,PNAME`*`)`
	`(`*`NRB,BUFFER,LENGTH,TIMEOU,PNAME,SSNM`*`)`
	`(`*`NRB,BUFFER,LENGTH,TIMEOU,PNAME,`*`NETX)`

SOFFR Parameters

The SOFFR call parameters are described below. The parameters must be specified in the order presented.

NRB

Points to the NRB data area to be passed to NetEx.

BUFFER

Points to the buffer data area to receive data sent by the corresponding SCONNECT request.

LENGTH

Specifies the length of the buffer (in addressable units or bytes for IBM) that holds the data sent by the corresponding SCONNECT. When called, length should contain the maximum size of the buffer. On return the NRBLEN (NRB.length) field contains the number of addressable units of information actually sent to the offering application. LENGTH is an integer.

TIMEOU

This required parameter is the number of seconds that the OFFER request should remain outstanding. If no application connects during this interval, then the OFFER ends abnormally. If TIMEOU is zero, the OFFER remains outstanding indefinitely.

PNAME

Specifies 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 packed array(.1..8.) of character, or as a string in the CALL statement if padded with blanks to eight characters in length (left justified blank filled).

SSNM

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

SOFFR Entry Parameters

The following NRB fields are used by SOFFR on entry.

BUFA

address for incoming data

BUFL

length of buffer to hold data

PROTA

address for incoming ODATA

PROTL
length of buffer to hold ODATA

TIME
number of seconds offer outstanding

CLASS
class of service requested

BLKO
maximum transmission size acceptable

BLKI
maximum reception size acceptable

MAXRT
limit on transmission speed

PNAME
application name to offer

SOFFR Results

The following NRB fields are updated when SOFFR completes.

STAT
success/failure code

IND
contains connect indication

LEN
length of incoming data

PROTL
length of incoming ODATA

UBIT
unused bit count of data

NREF
SREF assigned this connection

CLASS
class of service assigned

BLKO
maximum transmission data size

BLKI
maximum reception data size

MAXRT
maximum transmission speed of path

HNAME
name of host where SCONN originated

PNAME

name of application where SCONN originated

SCONN PASCAL 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.

SCONNW tells the calling program to wait for the call to complete before it resumes processing.

Operation (Select One)	**Required Parameters (Select One Group)**

`SCONN`
`SCONNW`
	`(`*`NRB,BUFFER,LENGTH,DATAMD,PNAME,HNAME`*`)`
	`(`*`NRB,BUFFER,LENGTH,DATAMD,PNAME,HNAME,SSNM`*`)`
	`(`*`NRB,BUFFER,LENGTH,DATAMD,PNAME,HNAME,`*`NETX)`

SCONN Parameters

The SCONN call parameters are described below. The parameters must be specified in the order presented.

NRB

Points to the NRB data area to be passed to NetEx.

BUFFER

Points to the buffer data area that holds the user data to be sent to the corresponding application.

LENGTH

Specifies the length of the data (in addressable units or bytes for IBM) 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. LENGTH is an integer.

DATAMD

Specifies the type of datamode to use when sending the connect data to the corresponding application. DATAMD is an integer. Refer to "NRBDMODE" on page 46 for a discussion of the DATAMD parameter.

PNAME

Specifies the alphabetic name of the process offered (SOFFR) by the corresponding calling program. The name offered is determined by the other calling program. This name may be provided as a packed array (.1..8.) of character, or as a string in the call invocation if padded with blanks to eight characters in length (left justified blank filled).

HNAME

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 packed array (.1..8.) of character, or provided as a string in the call invocation if padded with blanks to eight characters in length (left justified blank filled).

SSNM

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

SCONN Entry Parameters

The following NRB fields are used by SCONN on entry.

BUFA
address of outgoing data

LEN
length of outgoing data

PROTA
address of outgoing ODATA

PROTL
length of outgoing ODATA

UBIT
data unused bit count

DMODE
datamode of data

CLASS
class of service requested

BLKO
maximum transmission size acceptable

BLKI
maximum reception size acceptable

MAXRT
limit on transmission speed

HNAME
alphanumeric host name

PNAME
alphanumeric application name

SCONN Results

The following NRB fields are updated when SCONN completes.

STAT
success/failure code

LEN
set to zero

PROTL
set to zero

NREF
SREF (Session ID) assigned

CLASS
class of service assigned

BLKO
maximum transmission data size

BLKI
maximum reception data size

MAXRT
maximum transmission speed of path

SCONF PASCAL Call

The SCONF (confirm) call allows 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 an NRB with an NRBNREF relating to this session.

SCONFW tells the calling program to wait for the call to complete before it resumes processing.

Operation (Select One)	**Required Parameters**
`SCONF` `SCONFW`	`(`*`NRB,BUFFER,LENGTH,DATAMD`*`)`

SCONF Parameters

The SCONF call parameters are described below. The parameters must be specified in the order presented. All parameters are required.

NRB

Points to the NRB data area to be passed to NetEx.

BUFFER

Points to the buffer data area that holds the user data to be sent to the corresponding application.

LENGTH

Specifies the length of the data (in addressable units or bytes for IBM) 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. LENGTH is an integer.

DATAMD

Specifies the type of datamode to use when sending the connect data to the corresponding application. DATAMD is an integer. Refer to "NRBDMODE" on page 46 for a discussion of the DATAMD parameter.

SCONF Entry Parameters

The following NRB fields are used by SCONF on entry.

BUFA

address of outgoing data (move mode)

LEN

length of outgoing data

PROTA

address of outgoing ODATA (move mode)

PROTL

length of outgoing ODATA

UBIT

data unused bit count

DMODE

datamode of data

SCONF Results

The following NRB fields are updated when SCONF completes.

STAT
success/failure code

LEN
set to zero

PROTL
set to zero

SREAD PASCAL Call

The SREAD macro allows 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.

SREADW tells the calling program to wait for the call to complete before it resumes processing.

Operation (Select One)	Required Parameters
`SREAD` `SREADW`	`(`*`NRB,BUFFER,LENGTH,TIMEOU`*`)`

SREAD Parameters

The SREAD call parameters are described below. The parameters must be specified in the order presented.

NRB
Points to the NRB data area to be passed to NetEx.

BUFFER
Points to the buffer data area to receive the data sent by the corresponding application's SWRITE or SCONFIRM request.

LENGTH
Specifies the length of the buffer (in addressable units or bytes for IBM) 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 is in NRBLEN (NRB.length). Programs that wish to work with the Unused Bit Count on input should examine the NRBUBIT field (NRB.unusedbits). LENGTH is an integer.

TIMEOU
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 ends abnormally. If TIMEOU as zero, the READ remains outstanding indefinitely. The programmer should take alternate path retry into consideration when specifying the timeout value. Refer to "Alternate Path Retry" in the *H210IPZ NetEx/IP for IBM z/OS Installation Reference Manual.*

SREAD Entry Parameters

The following NRB fields are used by SREAD on entry.

BUFA
address for incoming data (move mode)

BUFL
length of buffer to hold data

PROTA
address for incoming ODATA (move mode)

PROTL
length of buffer to hold ODATA

TIME
number of seconds offer outstanding

SREAD Results

The following NRB fields are updated when SREAD completes.

STAT
success/failure code

IND
contains connect indication

LEN
length of incoming data

PROTL
length of incoming ODATA

UBIT
unused bit count of data

BLKO
maximum transmission data size (On Read of Confirm only)

BLKI
maximum reception data size (On Read of Confirm only)

SWRIT PASCAL Call

The SWRIT (write) call allows an application to transmit data to another calling program.

Before an SWRIT can be issued, a connection must be established.

SWRITW tells the calling program to wait for the call to complete before it resumes processing.

Operation (Select One)	Required Parameters
`SWRIT` `SWRITW`	`(`*`NRB,BUFFER,LENGTH,DATAMD`*`)`

SWRIT Parameters

The SWRIT call parameters are described below. The parameters must be specified in the order presented.

NRB

Points to the NRB data area to be passed to NetEx.

BUFFER

Points to the buffer data area that holds the user data to be sent to the corresponding application.

LENGTH

Specifies the length of the data (in addressable units or bytes for IBM) 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. LENGTH is an integer.

DATAMD

Specifies the type of datamode to use when sending the connect data to the corresponding application. DATAMD is an integer. Refer to “NRBDMODE” on page 46 for a discussion of the DATAMD parameter.

SWRIT Entry Parameters

The following NRB fields are used by SWRIT on entry.

BUFA

address of outgoing data

LEN

length of outgoing data

PROTA

address of outgoing ODATA

PROTL

length of outgoing ODATA

UBIT

data unused bit count

DMODE

datamode of data

SWRIT Results

The following NRB fields are updated when SWRIT completes.

STAT
    success/failure code

LEN
    set to zero

PROTL
    set to zero

SWAIT PASCAL Call

The SWAIT call waits for the completion of one of several NetEx requests before processing continues in the application. Control returns to the SWAIT caller if any of the specified NRBs complete the operation in progress. If any one of the NRBs has completed before the SWAIT call, control returns to the application immediately. Up to 16 SWAITs may be issued.

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

Operation	**Optional Parameters (Select One Group)**

`SWAIT`	`NRBNUM,NRB,NRB...[,SSNM]`
	`-1`
	`-1,SSNM`
	`-1,NETX`

SWAIT Parameters

The SWAIT call parameters are described below.

NRBNUM

Specifies the number of NRBs to wait for before processing continues. Control returns after any one of the specified calls/NRBs complete. If NRBNUM is positive, the corresponding number of nrb parameters must be supplied on this request.

NRB

Specifies a pointer to one or more NRBs (the number of NRBs specified in nrbnum) associated with the request.

-1

Specifies that the SWAIT caller wishes to wait for any NetEx request for this caller. No NRBs are supplied on this request.

SSNM

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

NOTE: Occasionally, an SWAIT-1 completes even though the user has acknowledged all NetEx requests. This may occur when NetEx marks event A completion, SWAIT-1 completes, control returns to the user, NetEx marks event B completion, user reviews all NRBs and finds that requests A and B have completed, user requests SWAIT-1 again, NetEx finds event B completion is marked, control returns to the user. To accommodate this sequence of events, the user program should be prepared to wait again.

SCLOS PASCAL Call

The SCLOS (close) call allows an application to transmit the last data to another calling program. If the other calling program has already issued an SCLOS, the session is gracefully disconnected.

Before an SCLOS can be issued, a connection must be established.

SCLOSW tells the calling program to wait for the call to complete before it resumes processing.

Operation (Select One)	**Required Parameters**
`SCLOS` `SCLOSW`	`(`*`NRB,BUFFER,LENGTH,DATAMD[,SSNM]`*`)`

SCLOS Parameters

The SCLOS call parameters are described below. The parameters must be specified in the order presented. All parameters are required.

NRB
: Points to the NRB data area to be passed to NetEx.

BUFFER
: Points to the buffer data area that holds the user data to be sent to the corresponding application.

LENGTH
: Specifies the length of the data (in addressable units or bytes for IBM) 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. LENGTH is an integer.

DATAMD
: Specifies the type of datamode to use when sending the connect data to the corresponding application. DATAMD is an integer. Refer to "NRBDMODE" on page 46 for a discussion of the DATAMD parameter.

SSNM
: This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

SCLOS Entry Parameters

The following NRB fields are used by SCLOS on entry.

BUFA
: address of outgoing data

LEN
: length of outgoing data

PROTA
: address of outgoing ODATA

PROTL
: length of outgoing ODATA

UBIT

data unused bit count

DMODE
datamode of data

SCLOS Results

The following NRB fields are updated when SCLOS completes.

STAT
success/failure code

PROTL
set to zero

SDISC PASCAL Call

The SDISC (disconnect) call allows 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, the user must wait for confirmation of previous SREAD or SWRIT calls before issuing the SDISC.

Before an SDISC can be issued, the user must provide a NRB with an NRBNREF relating to this session.

SDISCW tells the calling program to wait for the call to complete before it resumes processing.

Operation (Select One)	**Required Parameters**
`SDISC` `SDISCW`	`(`*`NRB,BUFFER,LENGTH,DATAMD[,SSNM]`*`)`

SDISC Parameters

The SDISC call parameters are described below. The parameters must be specified in the order presented.

NRB

Points to the NRB data area to be passed to NetEx.

BUFFER

Points to the buffer data area that holds the user data to be sent to the corresponding application.

NOTE: In the single case of SDISC, delivery of DISCONNECT data is not reliable, although the actual disconnection always occurs.

LENGTH

Specifies the length of the data (in addressable units or bytes for IBM) 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. LENGTH is an integer.

DATAMD

Specifies the type of datamode to use when sending the connect data to the corresponding application. DATAMD is an integer. Refer to "NRBDMODE" on page 46 for a discussion of the DATAMD parameter.

SSNM

This optional parameter is the subsystem (interface) name to use to access NetEx. The default is NETX.

Upon completion of the SDISC, the connection no longer exists. New requests against that connection are rejected. Issue an SOFFR or SCONN to establish a new connection.

SDISC Entry Parameters

The following NRB fields are used by SDISC on entry.

BUFA

address of outgoing data

LEN

length of outgoing data

PROTA

address of outgoing ODATA

PROTL
length of outgoing ODATA

UBIT
data unused bit count

DMODE
datamode of data

SDISC Results

The following NRB fields are updated when SDISC completes.

STAT
success/failure code

LEN
set to zero

PROTL
set to zero

COBOL High Level Programming Interface

NETEX includes a library of subroutines that are designed to be called by COBOL high level language programs. Also included are global data declarations and external reference declarations that may be included at compile time. This file is the *NRBCBL* member in the NTXMAC dataset. This member may be copied to any a user include/copy dataset. When the user makes a call to the user interface, the appropriate information is supplied in parameter format to pass to NETEX.

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 COBOL 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.

soffr	offer services
soffrw	offer services
sconn	connect to an offered program
sconnw	connect to an offered program
sconf	confirm acceptance of connect
sconfw	confirm acceptance of connect
sdisc	disconnect (immediate)
sdiscw	disconnect (immediate)
sread	read incoming request or data
sreadw	read incoming request or data
swrit	write data
swritw	write data
sclos	write last data
sclosw	write last data
swait	wait for previous request(s) to complete

These calls are described in “Session Layer Requests”.

COBOL NETEX Request Blocks

The C user creates an NRB by declaring a data structure of 21 fields using the NETEX supplied data structure template NRB. Various fields of this record will hold the information to be transferred to NETEX, and others will contain the information that is returned by NETEX. NRB variables should be declared to be of that type. Before these NRB records are used for any NETEX request, the caller is advised to zero all the elements of the record. This will allow defaults for fields not explicitly used by the caller to take effect.

The NETEX COBOL 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 field of the NRB record. When the request completes, the COBOL program directly accesses the desired fields of the NRB record to determine whether the operation completed properly. If NRB is declared as a 24 field control block, the fields of the block will be defined using the following:

Field	Type	Function
NRBSTAT	9(9) comp	Status returned from NETEX
NRBIND	9(9) comp	Data type indication from OFFER/READ
NRBLEN	9(9) comp	Length of data received in words
NRBUBIT	9(9) comp	Unused bit count
NRBREQ	9(9) comp	Request code
NRBNREF	char*	NETEX Reference number (N-ref) for this session
NRBBUFA	9(9) comp	User's Pdata buffer address
NRBBUFL	9(9) comp	Length of the buffer
NRBDMODE	9(9) comp	Datamode for WRITE request
NRBTIME	9(9) comp	Timeout for a read type request (in seconds)
NRBCLASS	9(9) comp	Class of service
NRBMAXRT	9(9) comp	Maximum rate of data transmission
NRBBLKI	9(9) comp	Blocksize to use for input
NRBBLKO	9(9) comp	Block size to use for output
NRBPROTA	char*	User's Odata buffer address
NRBPROTL	9(9) comp	Length of Odata buffer
NRBRESV1	9(9) comp	Reserved
NRBRESV2	9(9) comp	Reserved
NRBOFFER	x(8)	Offer Name (Source) - blank padded
NRBHOST	x(8)	Logical name of destination host - blank padded
NRBRESV3	9(9) comp	Reserved
NRBRESV4	9(9) comp	Reserved

NRBUSER	x(4)	Free for user to assign (nrbuser=nrbresv4)
NRBOSDEP	9(9) comp:16	Reserved for system dependent information

Figure 13. 'C' NETEX Request Block (NRB)

SOFFR COBOL Function

The SOFFR (offer) and SOFFRW (offer wait) functions make the services, provided by the calling NETEX application program, available to programs running on other hosts. The SOFFR is actually a specialized form of read request. The SOFFR reads any data associated with the SCONN.

Before issuing an SOFFR call, the user must provide an NRB (described in “NetEx Request Block”) to be used by the user interface.

SOFFR Function Format

The SOFFR function has the following format:

CALL (Select One)	Required Parameters

`CALL “SOFFR”`
`CALL “SOFFRW”`
	`USING nrb,buffer,length,timeout,pname[,ssnm].`

SOFFR Parameters

The following parameters were shown in the SOFFR function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters’ positions.

soffr
soffrw

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

nrb

(struct nrb *) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char *) This required parameter is the extended address of the buffer data area to receive data sent by the corresponding SCONNECT request.

length

(long:VALUE) This required parameter is the length of the buffer (in addressable units) 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 field will contain the number of bytes of information actually sent to the offering application.

timeout

(long:VALUE) This required parameter is 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 *timeout* is specified as zero, the OFFER will remain outstanding indefinitely.

pname

char*:REF:) This required 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 must be provided as a string in the CALL statement padded with blanks to eight characters in length.

ssnm

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

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
NRBPROTA	Address for incoming Odata
NRBPROTL	Length of buffer to hold Odata
NRBTIME	Number of seconds offer outstanding
NRBBLKO	Maximum transmission size acceptable
NRBBLKI	Maximum reception size acceptable
NRBMXRAT	Limit on transmission speed
NRBOFFER	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
NRBPROTL	Length of Odata received
NRBNREF	S-ref assigned this connection
NRBBLKO	Maximum transmission Pdata size
NRBBLKI	Maximum reception Pdata size
NRBMXRAT	Maximum transmission speed of path
NRBHOST	Name of host where S-conn originated

SCONN COBOL Function

The SCONN (connect) function provides a means for a program to request a session with a program that has issued an SOFFR. The SCONN is a specialized form of a write request. The SCONN initiates the session and may optionally write data to the offerer at the same time.

Before issuing the SCONN, an NRB must be provided for use by the user interface.

SCONN Function Format

The SCONN function has the following format:

CALL (Select One)	**Required Parameters**

`CALL “SCONN”`
`CALL “SCONNW”`
	`USING nrb,buffer,length,datamd,pname,hname[,ssnm].`

SCONN Parameters

The following parameters were shown in the SCONN function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters’ positions.

sconn
sconnw

This is the verb for this function. SCONNW specifies that the calling program must wait for the call to complete before processing is resumed.

nrb

(struct nrb*) This required parameter is the pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char*) This required parameter is the extended address of the buffer data area that holds the user data to be sent to the corresponding application.

length

(INT(32):VALUE) This required parameter is the length of the data (in addressable units) to be sent to the corresponding program. This value is 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.

datamd

(long:VALUE) This required parameter is the datamode to be used to send the connect data to the corresponding application. Refer to NRBDMODE in “NetEx Request Block” for a discussion of the datamode parameter.

pname

(char*REF:) This required parameter is the alphabetic name of the process offered (SOFFR) by the corresponding calling program. The name offered is determined by the other calling program. This name must be provided as a string in the call invocation, padded with blanks to eight characters in length.

hname

(char*REF:) This required parameter is 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 must be provided as a string in the call invocation, padded with blanks to eight characters in length.

ssnm

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

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
NRBPROTA	Address of outgoing Odata
NRBPROTL	Length of outgoing Odata
NRBBLKO	Maximum transmission size acceptable
NRBBLKI	Maximum reception size acceptable
NRBMXRAT	Limit on transmission speed
NRBHOST	Alphanumeric “host” name
NRBOFFER	Alphanumeric “application” name

SCONN Results

The following NRB fields are updated when SCONN completes.

NRBSTAT	Success/failure code
NRBNREF	S-ref (Session ID) assigned
NRBBLKO	Maximum transmission Pdata size
NRBBLKI	Maximum reception Pdata size
NRBMXRAT	Maximum transmission speed of path

SCONF COBOL Function

The SCONF (confirm) function 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.

The SCONF is a specialized form of write request. Data may be optionally written during the confirm process with this command.

Before issuing the SCONF function, 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.

SCONF Function Format

The SCONF function has the following format:

CALL (Select One)	**RequiredParameters**

`CALL “SCONF”`
`CALL “SCONFW”`
	`USING nrb,buffer,length,datamd[,ssnm].`

SCONF Parameters

The following parameters were shown in the SCONF function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters’ positions.

sconf
sconfw

This is the verb for this function. SCONFW specifies that the calling program must wait for the call to complete before processing is resumed.

nrb

(struct nrb*) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char*) This required parameter is the extended address of the buffer data area that holds the user data to be sent to the corresponding application.

length

(long:VALUE) This required parameter is the length of the data (in addressable units) to be sent to the corresponding program. This value is 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.

datamd

(long:VALUE) This required parameter is the datamode to be used to send the connect data to the corresponding application. Refer to NRBDMODE in “NetEx Request Block” for a discussion of the datamode parameter.

ssnm

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

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 C Function

The SREAD subroutine 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.

Important: Keep an SREAD request outstanding to receive incoming data. NETEX will automatically terminate a connection if a request is waiting to be read for too long. This read-timeout value is set at installation time.

Defaults for unspecified parameters are assumed to be the parameters existing in the NRB. After BUFFER and LENGTH are agreed on during the connection process, these parameters can be omitted.

SREAD Function Format

The SREAD function has the following format:

CALL (Select One)	Required Parameters

`CALL “SREAD”`
`CALL “SREADW”`
	`USING nrb,buffer,length,timeout[,ssnm].`

SREAD Parameters

The following parameters were shown in the SREAD function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters’ positions.

sread
sreadw

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

nrb

(struct nrb*) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char *) This required parameter is the extended address of the buffer data area to receive the data sent by the corresponding application’s SWRITE or SCONF request.

length

(long:VALUE) This required parameter is the length of the buffer (in addressable units) 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. Programs that wish to work with the Unused Bit Count on input should examine the NRBUBIT field.

timeout

(long:VALUE) This required parameter is 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 *timeout* is specified as zero, the READ will remain outstanding indefinitely.

ssnm

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

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
NRBPROTA	Address for incoming Odata

NRBPROTL	Length of buffer to hold Odata
NRBTIME	Number of seconds offer outstanding

SREAD Results

The following NRB fields are updated when SREAD completes.

NRBSTAT	Success/failure code
NRBIND	Contains Connect Indication
NRBLEN	Length of incoming Pdata
NRBUBIT	Unused bit count of Pdata
NRBPROTL	Length of Odata received
NRBBLKO	Maximum transmission Pdata size (On Read of Confirm only)
NRBBLKI	Maximum reception Pdata size (On Read of Confirm only)

SWRIT COBOL Function

The SWRIT (write) function provides a means for a program to transmit data to another calling program.

Before an SWRIT can be issued, a connection must be established.

SWRIT Function Format

The SWRIT function has the following format:

CALL (Select One)	Required Parameters

`CALL “SWRIT”`
`CALL “SWRITW”`
	`USING nrb,buffer,length,datamd[,ssnm].`

SWRIT Parameters

The following parameters were shown in the SWRIT function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters' positions.

swrit
swritw

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

nrb

(struct nrb *) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char *) This required parameter is the extended address of the buffer data area that holds the user data to be sent to the corresponding application.

length

(long:VALUE) This required parameter is the length of the data (in addressable units) to be sent to the corresponding application. The *length* may be set to zero.

datamd

(long:VALUE) This required parameter is the datamode to be used to send the connect data to the corresponding application. Refer to NRBDMODE in “NetEx Request Block” for a discussion of the datamode parameter.

ssnm

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

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 COBOL Function

The SCLOS (close) function provides a means for a program to transmit data to another corresponding calling program and indicates that this is the last data to be sent. The data must be received by a read request in the other program.

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

Before issuing the SCLOS, a connection must be fully established.

SCLOS Function Format

The SCLOS function has the following format:

CALL (Select One)	Required Parameters

`CALL “SCLOS”`
`CALL “SCLOSW”`
	`USING nrb,buffer,length,datamd[,ssnm].`

SCLOS Parameters

The following parameters were shown in the SCLOS function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters’ positions.

sclos
sclosw

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

nrb

(struct nrb*) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char*) This required parameter is the extended address of the pointer to the buffer data area that holds the user data to be sent to the corresponding application.

length

(long:VALUE) This required parameter is the length of the data (in addressable units) to be sent to the corresponding application. The *length* may be set to zero.

datamd

(long:VALUE) This required parameter is the datamode to be used to send the connect data to the corresponding application. Refer to NRBDMODE in “NetEx Request Block” for a discussion of the datamode parameter.

ssnm

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

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
NRBBUFA	Contains zero

SWAIT C Function

The SWAIT function provides a means to wait for NETEX completions that were issued "nowaited." The action taken by the subroutine will be different based on the value in the parameter NRBNUM.

If NRBNUM is greater than zero (swait specific) the subroutine will return control to the caller when any request identified in the NRB list has completed.

If NRBNUM is equal to zero, the subroutine will attempt to post all completed requests and return control to the caller immediately. If a previous call to SWAIT using NRBNUM equal to minus two (-2) has been issued but has not completed, the current SWAIT call will have no effect.

If NRBNUM is equal to minus one (-1), the subroutine will attempt to post all completed requests. If there were no completions at the time of the call, the subroutine will wait until a request has completed before returning control to the caller.

If NRBNUM is equal to minus two (-2), the action is system dependent.

SWAIT C Function Format

The SWAIT function has the following format:

CALL	**Parameters**
`CALL "SWAIT"`	`USING nrbnum,nrb,nrb...,nrb10[,ssnm].`
`CALL "SWAIT"`	`USING -2[,ssnm].`

SWAIT Parameters

The following parameters were shown in the SWAIT function format. The parameters must be specified in the order presented. If parameters are omitted, then commas must be used to preserve subsequent parameters' positions.

swait

This is the verb for this function.

nrbnum

(long:VALUE) This required parameter is the number of NRBs in the nrb or one of the values described above.

nrb

(struct nrb*) This required parameter is an address pointer to one or more NRBs (the number of NRBs specified in nrbnum) associated with the request to be waited for (maximum of 10). An *nrb* is required for each NRB specified in *nrbnum*.

ssnm

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

The SWAIT call provides the means to wait for the completion of requests that have not been previously waited for. Control will be returned to the SWAIT caller when 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. By waiting on 0 NRBs, the NETEX subroutine library will take control and update all NRBs, after which it will return control to the user.

After control is returned, the calling C program must determine which of the NRBs in the list has completed. This can be done by examining the NRBSTAT field of each of the NRBs.

SDISC COBOL Function

The SDISC (disconnect) function 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, the operator must wait for confirmation of previous SREAD or SWRIT calls before issuing the SDISC.

When an SDISC is issued, an outstanding SREAD in the other program will terminate with an error in NRBSTAT.

NETEX does not ensure that data written with an SDISC macro will actually be received by the other program.

SDISC Function Format

The SDISC function has the following format:

CALL (Select One)	**Required Parameters**

`CALL “SDISC”`
`CALL “SDISCW”`
	`USING nrb,buffer,length,datamd[,ssnm].`

SDISC Parameters

The following parameters were shown in the SDISC function format. The parameters must be specified in the order presented. If parameters are omitted, then the commas must be used to preserve subsequent parameters' positions.

sdisc
sdiscw

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

nrb

(struct nrb*) This required parameter is a pointer to the NRB data area to be passed to NETEX.

buffer

(extptr char*) This required parameter is the extended address of the buffer data area that holds the user data to be sent to the corresponding application.

Note: In the single case of SDISC, delivery of DISCONNECT data is NOT reliable, although the actual disconnection will always occur.

length

(long:VALUE) This required parameter is the length of the data (in addressable units) to be sent to the corresponding program. This value is 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.

datamd

(long:VALUE) This required parameter is the datamode to be used to send the disconnect data to the corresponding application. Refer to NRBDMODE in “NetEx Request Block” for a discussion of the datamode parameter.

On 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.

ssnm

Specifies the subsystem (interface) name to use to access NetEx. The default is NETX.

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

Appendix A. Asynchronous Post Exit Processing Routine

General

NetEx allows you to specify the address of a user-written routine that receives control after the conclusion of a NetEx request. This capability consists of interfacing with the MVS exit effectors, after the users NRB has been updated with the final status of a completed request. This interface to the exit effectors takes place from within the NXMSRB01 module, which executes as an SRB scheduled by the cross-memory data move processing routines. The asynchronous post exit requirement is communicated to NetEx by specifying the routine address as the parameter of the WAIT= address operand on a NetEx request. The MVS exit effector interface processing is as follows:

1. NetEx obtains an IQE/IRB pair through the CIRB macro to the stage one exit effector. This fills in some of the control block fields through operands of the CIRB macro.
2. NetEx maps pertinent data, defining the users exit requirements to the IQE/IRB.
3. NetEx BALRs to the stage two exit effector to schedule the users exit for execution. Input to the stage two exit effector is the IQE that defines the request along with the IRB. The IQE is queued to the ASXB. At this point, the users exit routine is ready for dispatch.
4. The MVS dispatcher invokes the stage three exit effector to queue the exit routines Interrupt Request Block (IRB) to the specified TCB. The IRB functions similar to the standard Program Request Block (PRB), except that it executes before any PRB. The next dispatch of the TCB executes the users exit unless a subsequent IRB has been scheduled for the same TCB.

NOTE: Neither the SRB routine nor the asynchronous post exit executes until the currently executing PRB relinquishes control to the system. However, either routine, if scheduled runs to completion before the PRB regains control.

If an MVS WAIT macro is issued before the asynchronous post exit receives control, the ECB address must be identified to NetEx through the ECB= operand. While an SRB can execute if a TCB is being WAITed on, the IRB executes under the TCB and does not receive control until the TCB WAIT is posted. Conversely, if an MVS WAIT macro is not issued, the ECB= operand of the request should be omitted entirely.

Appendix B. Glossary

This glossary contains terms that are used in this document. It includes frequently used acronyms and Network Executive Software product terminology.

ABEND
: Abnormal end.

ACB
: Access method control block.

ACK
: Acknowledge.

adapter
: A part that electronically or physically connects a device to a computer, another device or a communications line. A host interface connects to a computer, a link interface connects to a communications link.

address
: A unique identifier assigned to a device connected to a network.

alternate path
: Another NetEx network path an operation can use after a failure on a suspect path. See also alternate path retry (APR).

alternate path retry (APR)
: A facility that allows a failed operation to be retried on another NetEx network path from the device performing the I/O operation. Alternate path retry improves the operation of NetEx remote hosts by switching network activity from a suspect or failing path to an alternate network path automatically. Preferred path routing in NetEx/IP detects the recovery of a previously failing primary NetEx network path and automatically restores network activity to that path.

ALTIO
: Alternate I/O.

American Standard Code for Information Interchange (ASCII)
: A standard that defines the codes for a character set to be used for information interchange between equipments of various manufactures and is the standard for digital communications.

APAR
: Authorized program analysis report.

APF
: Advanced function printing.

APR
: Alternate path retry.

ASCII
: American Standard Code for Information Interchange.

Asynchronous
: A class of data transmission service whereby all requests for service contend for a pool of dynamically allocated bandwidth and response time.

ATTN
Attention.

BFX
Bulk file transfer.

BFXJS
Bulk file transfer job submitter.

BFXTI
Bulk file transfer initiater.

BFXTR
Bulk file transfer responder.

BIST
Built in self test. A set of comprehensive self tests built into numerous StorageTek products. The purpose of BIST is to verify as many aspects of a chassis as possible in a self-contained environment.

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.

buffer memory
Temporary storage areas. Data going to or coming from the chassis is temporarily stored here until transmission on the network is completed.

Bulk File Transfer (BFX)
A Network Executive Software application software package. BFX utility allows users of NetEx Communications software to move large quantities of data between similar or dissimilar types of processors on an IP network. Any one version of BFX is fully compatible with any other. BFX consists of three separate programs: BFX Transfer Initiater (BFXTI), BFX Transfer Responder (BFXTR) and BFX Job Submitter (BFXJS). BFX is a trademark of Network Executive Software.

channel
The channel subsystem facilities associated with a single channel path.

CMDPRE
Command prefix.

command list
A list of commands and statements designed to perform a specific task.

command prefix
A character that identifies a NetEx/IP subsystem. Use this character when issuing commands to a specific NetEx/IP subsystem.

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

control unit (CU)
A hardware unit that controls the reading, writing, or displaying of data for one or more input/output units.

CPU
Central processing unit.

CRC
Cyclic redundancy check.

CSW
Channel status word.

CTC
Channel-to-channel.

CTCA
Channel-to-channel adapter.

CTCB

CU
Control unit.

CUA
Common user access or channel unit address.

CUE
Control unit end.

cyclic redundancy check (CRC)
An error detection scheme.

DDB
Driver interface data block.

DEFBI
Default buffer input.

DEFBO
Default buffer output.

DREF
Driver reference.

EBCDIC
Extended binary-coded decimal interchange code.

EOF
End of frame.

ESTAE
Extended specify task abnormal exit.

ESTP
ESTAE parameter list.

FCB
Forms control block.

FCC
Federal Communications Commission.

firmware
A computer program or instruction that is used so frequently that it must be stored in a read-only memory instead of being included in software. Often used in computers that monitor production processes. Firmware may be marketed separately as PROMs.

FMID

Function module identifier.

frame

The unit of transmission on a Network Executive Software network. It consists of heading information and data. All data is sent on the network in this special envelope. A frame can be up to 4K bytes in length.

global network addressing (GNA)

A 32-bit network addressing scheme. It uses a 16-bit network address mated to the 16-bit unit address to uniquely identify units on a NetEx/IP tnetwork.

GNA

Global network addressing. A 32-bit network addressing scheme. It uses a 16-bit network address mated to the 16-bit unit address to uniquely identify units on a NetEx/IP network.

HCD

Hardware configuration definition.

header

Control information. It is 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.

host adapter

An interface connecting a host channel and a network. These interfaces include NESiGate.

host interface

An interface in a NESiGate that connects to a host channel.

IBM

International Business Machines.

ID

Identifier.

initial program load (IPL)

The process of starting (or restarting) the operating system.

I/O

Input/output.

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 the protocol stack (the layer concerned with routing datagrams from network to network).

IOCP

Input/output configuration program.

IPL

Initial program load.

IRB

Interrupt request block.

ISO
International standards organization.

JCL
Job control language.

JES, JES2, JES3
Job entry system.

JESCT
JES control table.

link
(1) A joining of any kind of networks. (2) May refer to the communications facility used to interconnect two local networks.

logical partition (LPAR)
A facility that allows you to divide the resources of a processor so that multiple copies of an operating system may exist on the same physical processor.

LPAR
Logical partition.

MCH
Machine check handler.

message proper
A basic addressing mechanism (nine to 64 bytes of data) used in all NetEx/IP interfaces. The first eight bytes are the message header and are common to all messages directed to the NetEx/IP network. These bytes indicate the source and destination addresses of both the physical adapters and the logical devices to which they are attached.

MIH
Missing interrupt handler.

MVS
Multiple virtual storage.

N/A
Not applicable.

NCT
Network configuration table.

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.

NETEX
NETwork EXecutive.

NETEX Request Block (NRB)
An array of parameters that pass information between calling programs and NetEx.

network
(1) A collection of interconnected computer systems, terminals and front-end processors. (2) Refers to the portion of a GNA address represented by the second byte (byte 1 when reading left to right). This portion of the GNA address has a hexadecimal value in the range from X’01’ to X’FF’. The

combination of domain, network, unit, and subaddress represent the GNA address of a particular processor on a network.

NETwork EXecutive (NetEx)

A Network Executive Software family of software designed to use IP networks that allows two or more application programs to communicate. NetEx is tailored to each supported operating system, but can communicate with any other supported NetEx, regardless of operating system.

NetEx can reside on the host, on a processor interface board, or in a NESiGate. The latter case uses host-resident drivers as interfaces. NetEx is a registered trademark of Network Executive Software.

NIT

NetEx internal task.

NRB

NetEx request block.

NREF

Network reference.

NUB

NetEx user block.

octet form

Information considered as a sequence of eight-bit bytes.

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.

path

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

PDS

Partitioned data set.

PFX

Print file transfer.

POST

Power on self test. One particular mode of BIST, POST is initiated a power-up, and runs on the chassis under test without operator intervention, special codes, or the connection of extraneous test equipment. POST tests all major logic boards in the chassis as quickly as possible, with a high degree of failure detection, and isolation to the board level.

port

(1) The point of entry/exit for data transmission to/from various chassis. (2) Ports may be logical or physical. The physical point of entry/exit for I/O operations to/from chassis is located on the I/O panel. (3) I/O operations through some chassis occur through separate logical data paths sharing a single physical port. Each of these data paths operates independently, as if it were a separate physical path. The point of entry/exit to these data paths is the logical port.

PRB

Program request block.

prefixed storage area (PSA)

An area of storage used for mapping fixed hardware and software locations per processor.

Print File Transfer (PFX)

A utility software package. The PFX allows users of NetEx communications software to transfer print files between similar or dissimilar types of processors on networks. PFX contains the facilities to select files and to make the file format conversions necessary for proper printing on the receiving host. PFX is a trademark of Network Executive Software.

protocol
A formal set of rules governing the format, timing, sequencing, and error control of exchanged messages on a network. A protocol may be oriented toward data transfer over an interface, between two logical units directly connected, or an end-to-end basis between two users over a large, complex network.

PSA
Prefixed storage area.

PSAREG
Prefixed storage area register save area.

PUT
Program update tape.

RAM
Random access memory. A memory device which may be written into or read from, with the time required to do so being independent of the data storage location. Information in a RAM is lost when the RAM loses power.

RU
Request unit check with bus out.

SID
Session identifier.

SLS
Satellite link subsystem.

SLU
Secondary logical unit.

SMF
System management facility.

SMP/E
System modification program extended.

SNA
System network architecture.

SRB
Service request block.

SREF
Session reference.

SSCVT
Subsystem communications vector table.

SSVT
Subsystem vector table.

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.

TREF
Transport reference.

TUB
Transport user block.

UCB
Unit control block.

UCW
Unit control word.

UIM
Unit information module.

unit
The portion of a GNA address represented by the third byte (byte 2 when reading left to right). This portion of the GNA address has a hexadecimal value in the range from X’01’ to X’FF’. The combination of domain, network, unit, and subaddress represent the GNA address of a particular processor on a network.

unit information module (UIM)
A software representation of a physical piece of hardware (for example, a control unit or a device) that describes the operating or connection rules for attaching that hardware to the processor for HCD.

WTO
Write to operator.

Index