FlexNet™
Network Operating System
User's Guide

1082-2003-001

First Edition: November 1986

Software Version: 1.3

FOREWORD

BEFORE YOU BEGIN

The presentation of information in this guide assumes you have already read the FlexOS™ User's Guide. Before you begin reading this guide, you should know:

- How to start FlexOS
- How to start a session on your computer
- How to use disks, directories, and printers
- How to enter and edit commands
- How to use the following FlexOS commands:

ASSIGN	Give a disk drive a new assignment
CHDIR	Change the current directory
COPY	Copy a file from one directory to another
DEFINE	Create a logical name
DIR	List the files in a directory
LOGON	Start a session on your computer
LOGOFF	End a session on your computer
MKDIR	Make a subdirectory
PATH	Define a search path

The FlexNet User's Guide describes how to:

- Start and end sessions on remote computers
- List the node names of computers on your network
- Use disks, files, and printers on other computers
- Display the network status
- Set up FlexNet. on the computer and change network parameters
- Use the print spooler system in the network environment

HOW THIS BOOK IS ORGANIZED

The FlexNet User's Guide consists of four sections and two appendices. Section 1 leads you through a brief exercise that demonstrates how to log on a remote computer and use its hard disk.

Section 2 provides general information regarding computer networks and an overview of FlexNet. Read this section for explanations of network terms and concepts and a description of how you use FlexNet.

Section 3 describes the FlexNet commands and lists the FlexOS commands you can use over the network.

Section 4 describes FlexNet installation. This section is intended for system managers.

Appendix A lists the network error messages.

Appendix B describes how to integrate nodes running PC DOS with nodes running FlexOS. Appendix B, like Section 4, is intended for system managers.

Appendix C explains print spooler use under FlexNet.

CONTENTS

Tables

Figures

Listings

GETTING STARTED WITH THE NETWORK

AN EXERCISE THAT USES THE NETWORK

A network is a combination of computer hardware and software that connects one computer to another. The network gives you access to disks and printers on the other computers. Computers on the network are referred to as nodes.

About This Lesson

The following exercise demonstrates how you can access a disk drive on another computer. The exercise is a sequence of commands and explanations.

To begin the exercise, turn on your computer and, if required, enter your FlexOS LOGON command. The operating system's prompt must be displayed in order for you to enter the commands that follow.

List the Nodes

Enter the command

```
C>NAMES LIST
```

This command displays your node's name and the names of the nodes and sockets on your network. Node names identify the computers on the network. You designate which computer you want access to by using its node name. You do not use sockets directly; the NAMES LIST command displays this information for the system manager's use.

You will need to use a node name when you enter the commands that follow. Pick one of the names under the **Remote Node(s)** heading in the NAMES LIST display. Use the node name you pick in place of `node::` as you enter the commands.

Log on a Server

Enter the command

```
C>LOGON node::
```

Enter your user name and password when LOGON prompts you. The system prompt is returned to your screen when LOGON completes.

Just as you use the LOGON command to start a session on your own computer, you also use it to start sessions on another computer. In this manual we'll refer to starting a session as logging on. Ending a session is called logging off.

The computers on the network you can log on are called "servers." The disk drives and printers available on a server are called "remote resources." The computer you are sitting at is referred to as the "local" computer.

You log on servers one at a time. If your network has more than one server, you can be logged on to multiple servers simultaneously.

Display Remote Files

Enter the command

```
C>DIR node::C:
```

This command displays the contents of drive C: on the server you logged on.

You use the DIR command to display the files in the server's directories. To distinguish a server's drive from a drive on the local computer, precede the drive specification with the server's node name.

Give the Remote Drive a Logical Name

Enter the command

```
C>DEFINE G:=node::C:\
```

This defines the logical name G: as the server's drive C:. Now you can use G: to designate this drive in your commands instead of the full network path. For example, enter DIR G: and notice that you get the same directory display shown by the previous command.

A full network path consists of a node name, drive, directory, and filename.

Show Network Status

Enter the command

```
C>NETSTAT
```

NETSTAT is the FlexNet network status command. You can use NETSTAT to determine which servers you have logged on. See Section 3 for the complete description of NETSTAT and its options.

Summary

In this exercise you displayed the names of the nodes and sockets on your network, logged on a server, displayed the files in the root directory of that server node's drive C:, and defined a logical name for that drive. These steps demonstrate three general rules for using the network:

- You must log on a remote computer before you can use any of its resources.
- You designate a remote device by using the remote computer's node name before the device name.
- You use standard FlexOS commands to access server resources.

The next section describes more of the how, when, and where of logging on and remote resource use. Section 3 describes the network commands.

End of Section 1

WHO'S WHO ON THE NETWORK

The network allows you and other users to share disks, printers, and files among different computers. Although some networks give you access to every user's disks and printers, most have security measures that limit access or deny it altogether to some of the nodes. This section describes who is who on the network and which remote resources are available to you.

Note: There are few hard and fast rules governing network configuration (that is, which computers are connected to each other and who can access whom). In fact, one of FlexNet's features is its configuration flexibility--its ability to be reconfigured as more users, computers, and resources are needed. Consequently, we cannot describe exactly which nodes on your network are servers and what resources are available on them. For explicit information on your network's nodes and remote resources, see your system manager.

NODES

Computers on a network are called nodes. Each node is identified by a unique name and is either a requester-only or a combination requester and server. Figure 2-1 illustrates a network with both types of nodes.

From requester-only node computer, you can run your programs and send requests to other computers.

Here are some examples of the requests you can make:

- Log on another computer
- Display the files in a remote directory
- Print a file
- Make a new directory
- Copy a file
- Get data from a file

A requester-only node's drives and printers are not available to other network users.

Combination requester and server nodes (henceforth called requester/servers) are computers from which you can issue requests while simultaneously the computer services requests from other users. The requester and server mechanisms are completely separate. If your computer is a combination requester and server, FlexNet and FlexOS complete remote requests without interrupting your program.

Networks can be configured with requesters and requester/servers in any combination. For example, the network shown in Figure 2-1 is composed of three requester nodes and two requester/servers; the five nodes in Figure 2-1 could also configured as five requester/servers or as one requester/server and four requesters.

Each node has a name. Use the node name in your commands to designate which node you want to use. For example, Joe in Figure 2-1 enters

```
DIR SRVR::C:
```

to display the contents of drive C: on node SRVR::.

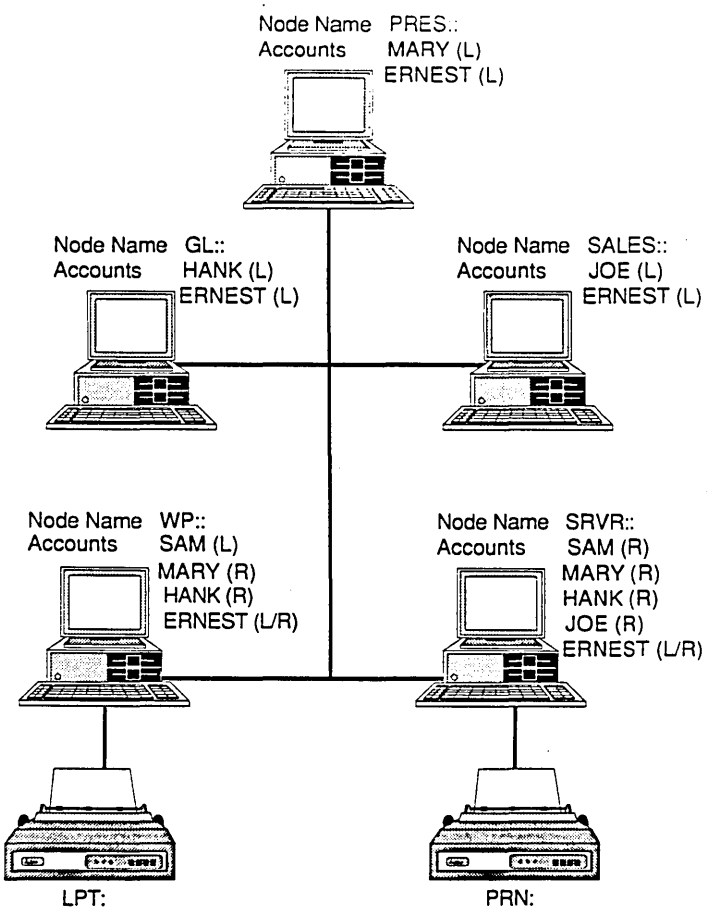


Figure 2-1. A Sample Network

This network has two requester/server nodes, SRVR:: and WP::, and three requester-only nodes. Each user has an account on SRVR::. Only SAM, MARY, HANK, and ERNEST have accounts on WP::; JOE cannot use this node. The account mode determines whether the user can access the computer locally (L), only from a remote computer (R), or both locally and remotely (L/R).

LOGGING ON

Logging on means starting a session on a computer. To log on your computer, enter the LOGON command and enter your account name and password. The LOGON program makes sure you have an account on the computer and confirms that the password is correct and returns the system prompt. You can then proceed to use the computer's disks and printers.

You must also log on a remote node to use its disks and printers. This establishes a connection (like a telephone connection) through the network between your node and the remote node. The connection remains in place until you log off.

You must have an account on the remote node to log on. However, an explicit LOGON command might not be necessary. Whether or not LOGON is required is determined by the server security mode.

Note: The description of accounts, access modes, and server security applies only to nodes on which the FlexOS protection mechanism is installed. If your network has no nodes with protection (your system manager can tell you if it does or does not), you can skip to "Shared Resources" below.

Nodes You Can and Cannot Log On

The network allows you to log on requester/server nodes from another node. You must have a user account on the node, however, or FlexNet does not allow you to log on.

You cannot log on requester-only nodes from other nodes.

User Accounts and Their Access Mode

Each computer has a user account file. Each account entry in the file contains a user name, password, user and group identification numbers, home directory, and access mode. The LOGON command checks the name and password you enter against the entries in the file. If there is a match you are allowed to log on.

You can log on under one account at a time; you cannot log on two accounts simultaneously. You do not need to log off explicitly, however, to change accounts. To log on under a different account, run the LOGON command and specify a different account name. FlexNet automatically logs you off the other account.

The access mode determines whether you can log on as a local user, a remote user, or as either a local or remote user. Users with accounts that allow local access only cannot log on the node from another. They can, however, log on from the computer's keyboard. Users with accounts that allow remote access only cannot log on from the computer's keyboard. They can, however, log on from another network node. Users with accounts that allow local and remote access can log on from the computer's keyboard or another node.

Server Security Modes

There are two server security modes: normal or extended. Requester/server nodes with normal security do not require an explicit LOGON command to log on from a remote node. Instead, FlexNet logs you on automatically the first time you put the node name in a path specification. For example, if WP:: in Figure 2-1 above has normal security, Hank is logged on automatically by the command

```
DIR WP::C:
```

To take advantage of this convenience, you must have an account on the requester/server and the remote account's name must match your local account name. If the account names are different, you must log on the requester/server with the LOGON command regardless of its security mode.

Requester/server nodes with extended security require you to log on with the LOGON command at all times. Section 3 explains how to use the LOGON command.

The security mode of one requester/server is independent of other nodes. You can have one requester/server on the network with normal security while another has extended security. Your system manager can tell you whether a requester/server has extended or normal security.

User Accounts and Access Modes Illustrated

Figure 2-1 above shows how accounts and access modes are used to determine who can log on which nodes. Ernest is the system manager of this network and has given himself an account on each node.

Mary, Hank, and Joe have their own computers (nodes PRES::, GL::, and SALES::, respectively) with accounts set for local access mode. These users can log on their computers directly (that is, from the keyboard), but they cannot log on from another node.

The node SRVR:: has an account for all users. Each account except for Ernest's has the remote security mode. Ernest's has the local and remote mode. This means that Mary, Hank, Joe, and SAM can log on SRVR:: from their own computer. However, only Ernest can log on SRVR:: from its keyboard.

The WP:: node is also shared by several users. However, it also has its own user--Sam. Only Sam's and Ernest's

accounts allow a local log on. Mary and Hank can only log on from their own nodes. Joe cannot log on the WP:: node at all.

Networks can have some nodes with protection installed and some without. See your system manager for answers to questions regarding local and remote protection, accounts, and access modes.

SHARED RESOURCES

When you log on a FlexOS requester/server node, you have access to the root directory on all disks and all printers on that computer. Access to subdirectories and the files in them depends on whether the protection mechanism is installed. See "IDs and Access Privileges" below for the description of the protection mechanism and how it affects file access. If file protection is not installed, you have access to all files and printers.

Remote File Specification

To access a file on another node, put the node name at the front of the path specification. For example, to type the file AR.DAT with the path C:GL\ACCOUNTING on node SRVR::, enter:

```
TYPE SRVR::C:\GL\ACCOUNTING\AR.DAT
```

You can also use the node name in a logical name assignment:

```
DEFINE DATA=SRVR::C:\GL\ACCOUNTING\AR.DAT
```

"FlexOS Commands" in Section 3 has more examples of node name use in path specifications.

Remote Printer Specification

To specify a remote printer, use the same form as a remote drive and file specification by preceding the printer name with the node name. For example, the path that specifies a printer named LPT: on node WP:: is:

WP::LPT:

You could give this printer the logical name REMPRINT: with the command

DEFINE REMPRINT:=WP::LPT:

IDs and Access Privileges

On nodes where the FlexOS file protection mechanism is installed, you might be restricted from accessing some subdirectories and files. The system manager assigns each user a user identification number (user ID) and a group identification number (group ID). FlexOS uses these numbers to deny some users and allow others access to a file or a directory.

Your user ID and group ID are a part of your account. Your user number is unique. Your group number is shared with others who use the same data and programs. For example, all the users in an accounting department would have the same group number. You can have different user and group identification numbers on different nodes.

Every file and directory is recorded on the disk with the user and group identification number of the user who created it (referred to as the file owner). In addition, the access privileges for the owner, the group, and everybody else (referred to as the world) is recorded as well.

There are four access privileges.

- **READ:** The user is allowed to read the file.
- **WRITE:** The user is allowed to read the file and modify its contents.
- **EXECUTE:** The user can run the file. (This privilege applies to program files only).
- **DELETE:** The user can delete the file and change its file attributes.

The access privileges for the owner, group, and world are established when the file is created If you have the DELETE privilege, you can change the access privileges with the FSET command. The FSET command also allows you to change file attributes; see the FlexOS User's Guide for a description of FSET.

Access privileges determine whether or not you can use a file or directory. As the file owner, you would typically have all four privileges shown above. As a group member, your privileges might be reduced to READ and EXECUTE only; you would not be able to delete the file or modify its contents. As a world member, your privileges might be further reduced to READ or you may have no privileges at all. Your system manager can fill you in on what your owner, group, and world access privileges are for different files.

PC DOS VERSUS FLEXOS NODES

So far, this description of network use assumed your network is composed entirely of computers running FlexOS. Your network can also include requester and server nodes running IBM® PC DOS and the PC Local Area Network Program (PC LAN). Appendix B contains PC DOS node information for system managers.

If your network consists solely of FlexOS, skip to "You and the System Manager," below.

Using a PC DOS Server from a FlexOS Requester

For the most part, logging on and using a PC DOS server is just like using a FlexOS server. The differences are as follows:

- Instead of the node and all its resources, you log on a single device on a PC DOS server.
- You enter the device's shortname instead of an account name in the LOGON command. You cannot use the device's path to log on; you must used the shortname assigned to it with the PC DOS NET SHARE command.
- Logging on more than one device does not log you off devices previously logged on.

You use the LOGON command to log on a PC DOS server. In the command, use the FlexOS node naming convention (nodename::) rather than PC DOS's (\\nodename). When prompted for the user name, enter the shortname assigned to the device.

If the device is protected with a password, LOGON will prompt you for it.

Using a FlexOS Server from a PC DOS Requester

Use the standard PC DOS NET USE command and syntax to log on a FlexOS server from a PC DOS node. Note, however, that you must use a shortname for a directory path rather than the path specification itself.

FlexOS interprets the shortname as the logical name for a directory path. FlexOS limits you to that directory and its subdirectories. FlexNet allows you to log on multiple directories on the same node simultaneously from a PC DOS requester.

Unless the FlexNet spooling system is installed and running in your network, your access to a FlexOS node's resources from a PC DOS requester is limited to disks and directories. Without the spooling system, any printers on the FlexOS node are inaccessible. See your system manager for details about the spooling system.

YOU AND THE SYSTEM MANAGER

Because each network is different, we have not tried to describe the specifics of your network. For example, FlexNet is equally suited for small networks consisting of a few requester/server nodes with no protection and large networks with complex configurations of requester/servers, requesters, accounts, and access privileges.

Your system manager can give you specific information on your system configuration. The system manager set up the network by defining node names, installing protection where necessary, putting in the accounts and access modes on each node, and assigning the user and group numbers.

See your system manager to find out which nodes you have

accounts on, who is your primary server, which directories are available to you on each node, which servers have normal versus extended security, and what programs you can use on each node. The system manager can also help you when you need changes made.

End of Section 2

NETWORK COMMANDS

This section is divided into three parts:

- "FlexNet Commands" describes the FlexNet commands.
- "FlexOS Commands" on page 3-13 identifies which FlexOS commands to use for access to remote FlexOS and PC DOS nodes.
- "PC DOS Commands" on page 3-15 lists the PC DOS commands you can use to access resources on FlexOS nodes and describes general rules for their use.

To access resources on FlexOS and PC DOS servers from FlexOS requesters, use the standard FlexOS commands. To access resources on a FlexOS server from a PC DOS requester, use the PC DOS commands. Never run a PC DOS command on a FlexOS node or a FlexOS command on a PC DOS node.

FLEXNET COMMANDS

Three network-specific commands are provided with FlexNet:

NAMES: sets and displays node names
NETSTAT: displays network status information
NETSET: sets network parameters

NAMES and NETSTAT are described in this section. NETSET is described in Section 4. Descriptions of how to use the FlexOS LOGON and LOGOFF commands with node names and accounts are also included in this section. Command descriptions are presented in alphabetical order.

LOGOFF

Forms:

(1) LOGOFF
(2) LOGOFF node::
(3) LOGOFF node::shortname

Explanation: LOGOFF ends a session on a computer. To access the computer again, you must log on first.

Form 1 ends the session on your local computer and all remote computers. To use the computer, you must log on again.

Form 2 logs off the remote computer specified by node::. All other sessions with other computers remain active. If the node specified is a PC DOS server, this command logs off all devices and directories on that server.

Form 3 logs off the device or directory specified by shortname on the remote computer specified by node::. Use this form to log off a single directory or device on a PC DOS server without logging off other devices.

LOGON

Forms:

(1) LOGON
(2) LOGON node::
(3) LOGON node::username
(4) LOGON node::shortname

Explanation: LOGON starts a session on a computer. You must have a user account on the computer and the account password to start the session.

Use form 1 to log on your computer. Use form 2 to log on a remote computer. After you have entered the command, LOGON prompts you to enter your user name (also called an account name) and your password. You get the system prompt when you log on your computer. You get a success message indicating the remote node and user name when you log on a remote computer.

Use form 3 to specify the remote node and account name in the LOGON command line. LOGON prompts you to enter the password.

Use form 4 to log on PC DOS servers. Use the FlexOS node syntax (node::) rather than the PC DOS syntax (\\node) in the LOGON command. You must use the shortname to designate the device; you cannot use the path specification. (See the NET SHARE command

description in the PC Local Area Network Program User's Guide for instructions on assigning shortnames and device passwords.) FlexOS prompts you for the device password.

You log on FlexOS requester/servers one account at a time. If you log on the same node again under a different user name, DR Net logs off the previous account.

You can log on PC DOS servers multiple times without affecting previous logons. This lets you log on several of the node's directories and devices simultaneously.

Note: LOGON is only necessary to log on PC DOS servers and FlexOS requester/servers with extended security. Logging on is automatic on FlexOS requester/servers with normal security where your account's user names matches your local user name.

NAMES

Forms:

(1) NAMES HELP
(2) NAMES SET LOCAL nodename
(3) NAMES GET LOCAL
(4) NAMES LIST
(5) NAMES PARSE filename

Explanation: NAMES manages your directory of nodes. There are four basic directory management tasks:

- Setting your node's name
- Getting your node's name
- Listing the nodes in your network
- Checking the names database for form errors

Use form 1 to display a brief explanation of the command options.

Use form 2 to set the node's name. In most cases, your node's name is defined for you in the configuration program that loads FlexOS. You cannot change your node's name once it has been set.

Use form 3 to find out the name of your node. Note, however, that your node name is shown in the NAMES LIST (form 4) display.

Use form 4 to list all of the names of the nodes on your network. To get a subset of the names, enter a

partial node name with the *, ?, and ^ wildcard characters. Figure 3-1 illustrates the NAMES LIST display for the sample network in shown Figure 2-1 as Mary would see it.

Note: The NAMES LIST command also lists the network sockets. Sockets are connection points on each node. This information is valuable to the system manager but has little if any use to users.

Use form 5 to test the NAMES database file for format errors. See Section 4 for the description of the NAMES database file and the use of this command.

Examples: The following example of NAMES form 2 sets the local node name to MARK::.

```
A>NAMES SET LOCAL MARK
```

The following example of form 3 lists all nodes whose name starts with MA.

```
A>NAMES LIST MA*
```

The following example of form 5 checks each name entry in the STDNAMES.DAT file for format errors.

```
A>NAMES PARSE STDNAMES.DAT
```

```
LIST OF NETWORK NAMES BY NAME TYPE
----------------------------------

Local Name:                PRES

Remote Node(s)             GL
                           SALES
                           WP
                           SRVR

Local Socket(s)            DRIRQSTR
                           DRISRVR
                           DRINAMES

Remote Socket(s)           None known
```

Figure 3-1. Sample NAMES LIST Display

NETSTAT

Forms:

(1) NETSTAT HELP
(2) NETSTAT
(3) NETSTAT E
(4) NETSTAT A
(5) NETSTAT [opt]C

Explanation: NETSTAT displays network connection status. A connection is a virtual circuit between two nodes. You make a connection when you log on another node. Network applications also make connections. Use form 1 to display a NETSTAT option summary.

Form 2 lists your connections. The NETSTAT display lists the remote nodes' name, your account name on that server, and the connection status. Table 3-1 lists the connection states.

Use form 3 to display the nodes you are logged on, how many bytes of data have been transferred through the connection, and your user ID and group ID on each node. NETSTAT formats this information as shown in the top half of Figure 3-2.

Form 4 lists the nodes you have logged on and, if your node is also a server, all remote nodes logged on to your node. Figure 3-2 illustrates the NETSTAT A display.

Form 5 runs NETSTAT at fixed intervals. NETSTAT prompts you to specify the interval. The time unit is seconds. Enter A or E for [opt]. For example, the following command repeats the NETSTAT A display:

```
NETSTAT AC
```

The information on the message bytes received and sent is cumulative starting from the time connection was established. This NETSTAT information is useful for network development and monitoring, but has little application for day-to-day network use.

The LISTEN entry under the Status header in Figure 3-2 indicates that the FlexNet server mechanism is waiting for a logon request from another node. All requester/server nodes have a LISTEN entry unless they have reached their maximum number of remote logons.

```
                    Network Status Utility

You are connected to these servers:

Node            User    Status   Received   Sent       Uid  Gid
--------------- ------- -------- ---------- ---------- ---- ----

These nodes are using your computer:

Node            Status   Received   Sent
--------------- -------- ---------- ----------
                LISTEN   0
```

Figure 3-2. NETSTAT A Display

The NETSTAT A display shows the nodes you logged on and the nodes logged on to your computer. "Received" and "Sent" indicate the number of message bytes transferred since the connection was opened. "Uid" and "Gid" indicate the user ID and group ID for the account on that node, not the local node. The NETSTAT E display shows the server information alone.

Table 3-1. NETSTAT Connection States

Status	Description
IDLE	The socket has no connections.
LISTEN	FlexNet server mechanism is waiting for a connection request to come in.
CONNECT	You have requested a logon, but FlexNet has not finished processing your request.
VERIFY	FlexNet has established a connection between the two sockets. However, FlexNet has not determined whether or not your node and the other speak the same network dialect.
OPEN	You have successfully logged on the server and can communicate with it.
CLOSING	You have logged off but FlexNet is not through processing the request.

Table 3-1. Cont'd

Status	Description
REOPEN	You have requested a logon after a logoff, but FlexNet has not yet logged you off.
FAIL	An error has been detected; this connection is no longer reliable.
FLUSH	Connection requests have failed and all connection processing has been flushed.
DIALURE	A dialect failure occurred during VERIFY.
XWAIT	FlexNet is waiting for a transport operation to complete.

FLEXOS COMMANDS

Use the standard FlexOS commands to access remote files and printers over the network. Some of the commands, however, can have disasterous consequences if misused. The following commands are safe for general use.

ASSIGN	DIR	MKDIR	RESTORE
BACKUP	ERASE	MORE	RMDIR
BATCH	EXIT	PATH	SORT
CHDIR	FIND	PRINT	TREE
COMP	FSET	RECDIR	TYPE
COPY	LOGOFF	RECFILE	VOL
DEFINE	LOGON	RENAME	

Specify the remote computer, rather than the local computer, by placing the node name at the beginning of the path. For example, enter the following command to make a directory called PRIVATE on node SRVR:: drive C:

```
MD SRVR::C:/PRIVATE
```

To copy all files from your default directory to the new directory, use the node name in your COPY command as follows:

```
COPY *.* SRVR::C:/PRIVATE
```

You can also use the node name in a logical name definition. The following DEFINE command assigns the name MYDIR: to the directory just made:

```
DEFINE MYDIR:=SRVR::C:/PRIVATE/
```

You also use the node name in a file's directory path. For example, to display a file named RESUME.ME in the PRIVATE directory one screen-full at a time, you could enter the following command:

```
MORE < SRVR::C:/PRIVATE/RESUME.MS
```

Using the logical name previously defined, the same command would appear as shown below.

```
MORE < MYDIR:RESUME.ME
```

The following FlexOS commands should be used with extreme caution or eliminated from network use entirely:

CHKDSK	FDISK
DISKCOMP	FORMAT
DISKCOPY	SYS
DISKSET	

Generally, these commands are useful only to the system manager.

Use the following FlexOS commands to access files on PC DOS servers:

BATCH	DIR	MKDIR	RMDIR
CHDIR	ERASE	MORE	SORT
COMP	FIND	PATH	TREE
COPY	LOGON	PRINT	TYPE
DEFINE	LOGOFF	RENAME	VOL

In all cases, redirection to or from a PC DOS server file is supported. Note that PC DOS supports only the * and ? wildcard characters; only FlexOS nodes support the ^ wildcard. Be sure to use the FlexOS node naming convention in your path specification rather than the PC DOS convention.

PC DOS COMMANDS

On PC DOS nodes, use PC DOS commands to manage files and directories on FlexOS servers. The FlexNet print spooler must be installed on a FlexOS server node before you can access its printer from a PC DOS requester. You cannot access printers on FlexOS servers from a PC DOS requester using the PC DOS PRINT command, the PrtSc function key, or the PC LAN NET PRINT command.

You can use the PC DOS commands listed below. Be sure to use the PC DOS node naming convention, rather than the FlexOS convention, in your path specifications.

APPEND	DEL	RENAME
ASSIGN	DIR	RESTORE
ATTRIB[1]	ERASE	RMDIR
BACKUP	EXE2BIN	SORT
BATCH	FIND	SUBST
CHDIR	JOIN	TREE
COMP	MKDIR	TYPE
COPY[2]	MORE	VOL

1. The only person who can change the read-only attribute on a FlexOS server is the person who created the file.
2. You can copy files to a remote disk. You can copy files to a remote printer on a FlexOS server if the spooler is installed.

Use the NET USE command to log on a FlexOS server. Use of the other NET commands is not affected by a FlexOS server except for NET PRINT, NET FORWARD, and NET SEND. For these commands, the destination node cannot be a FlexOS server.

End of Section 3

NETWORK INSTALLATION

FlexNet software consists of driver modules, command files, and data resource files. This section describes how to select the appropriate drivers and commands for each node and how to create the data resource files.

Note: Before proceeding with FlexNet installation, you should have the network hardware and cables installed. In addition, make a copy of each FlexNet distribution diskette and use the backup instead of the original. Store the originals in a safe place.

FlexNet installation consists of the following tasks:

1. **Make a general plan:** Determine which computers are servers and which are requesters.
2. **Create a name service database:** Make the NETNAMES.DAT file for each node.
3. **Install accounts:** Add remote users to the USER.TAB file on each server node.
4. **Set network parameters:** Create a NETSET resource file for each node that requires changes to default values.
5. **Update each node's CONFIG.BAT file:** Add DVRLOAD commands to install the required drivers, make required changes with the NETSET command, and set the local node name.
6. **Record FlexNet files:** Copy the required drivers and command files to the computer's system: drive.
7. **Load applications:** Record the program and data files for the applications to be shared on the requester/servers.

8. **Test the network:** Make sure the accounts and programs are installed properly.

This section describes each task in the order listed above. If you have need to integrate servers or requesters running the IBM PC Local Area Network Program, see Appendix B for the installation instructions after reading this section.

MAKE A GENERAL PLAN

Note: Some networks impose restraints on the configuration of requesters and requester/servers. Ask your network vendor if there are restraints and how they can affect your configuration.

Your plan is the network configuration blueprint. The complete plan indicates which computers are requester/server nodes, which are requesters only and which users have accounts on which nodes. Keep in mind these goals as you develop your plan:

- **Optimize use of hard disks and printers:** Put hard disks and printers on requester/server nodes so more than one user can access them.
- **Balance the load on requester/servers:** Too many requesters on the same requester/server impedes program and network performance.
- **Provide security where confidential data must be protected:** Enable protection on server nodes that have programs and files that must be restricted from general access. Assign user and group identification numbers and set access privileges to limit file and program access.
- **Minimize file duplication by sharing applications and data:** Distribute access to programs and data used by

many requesters. This reduces file storage requirements, eases file maintenance and updating, and ensures that all users have the most up-to-date data. Keep the following points in mind while you consider which applications will have shared access:

- It is illegal to run some applications on more than one computer.
- Some applications do not have locking mechanisms to prevent two users from changing the same file record simultaneously. This can have disasterous consequences.

Gather all the information you need to complete the plan before you go on to the next step. To help in this task, make a map of your network and develop a summary form for each node. Figure 4-1 illustrates a form that contains the important node characteristics.

Most of the items shown in the form are explained in this manual or in the FlexOS User's Guide. Two new network characteristics introduced here are "node addresses" and "sockets."

Node addresses are the physical addresses used by the network hardware to identify each node. Each node has a different address. Get the address values from the network hardware or your network vendor.

Users should never use the node address to designate a node. They must always use the node name. It is your task to name each node and provide FlexNet with the name-to-address translation in a file named NETNAMES.DAT. NETNAMES.DAT is described later in this section.

Sockets are logical plugs used by FlexNet to make and distinguish connections between nodes. When you log on another node, FlexNet makes a virtual connection between a socket on your node and a socket on the requester/server--somewhat like the connections operators made on old-style telephone switchboards to connect one caller to another.

FlexNet use two sockets: DRIRQSTR on each requester and DRISRVR on each requester/server. Network applications, such as an electronic mail package, use other sockets to distinguish their connections from those used by FlexNet and other applications. You can have multiple connections to different nodes out of the same socket. For example, all connections to a FlexOS requester/server come into the DRISRVR socket.

Sockets, like nodes, are identified by a name and an address. The socket name is defined by the program that uses it (you do not make up names for sockets). You provide FlexNet with the socket's name-to-address translation in the NETNAMES.DAT file along with the node name translation.

Note: The addresses for DRIRQSTR, DRISRVR, and a third reserved socket, DRINAMES, are set within FlexNet. You do not need to provide name-to-address translations for these sockets.

Before proceeding with the next section, review the manuals for your network applications to determine if they use any sockets and, if so, make a list of all the socket names. Get the addresses for these sockets from your network vendor.

Node

Name	Address	Type	Extended security?

Resources

Disk Drives	Printers

Sockets

Name	Address

Accounts

User Name	User ID	Group ID

Programs

Name	Directory	Owner	Group	Security

Data Files

Name	Directory	Owner	Group	Security

Figure 4-1. Sample Node Form

CREATE NAME SERVICE DATABASE

FlexNet requires you to store the node and socket translation information for each node in a file named NETNAMES.DAT. Each node must have its own NETNAMES.DAT file recorded on its system: drive with the system (-S) attribute.

Create NETNAMES.DAT with an editor or word processor. Each node and socket name entry in the file must adhere to a specific format. Figure 4-2 illustrates the entry format.

Note: If you use a word processor, be sure that it does not insert special characters in the file. (Some word processors use special or control characters to justify lines, change typestyles, underscore, and so forth.)

Each NETNAMES.DAT entry consists of the field identifiers NAME:, ADDRESS:, TYPE:, and, optionally, OEM:. Terminate each NAME:, ADDRESS:, and TYPE: entry with a Carriage Return. The OEM: field is an implementation option. Do not include the OEM: field identifier unless your network vendor has instructed you to.

You can place comments on any line in the file. Comments must be delimited by /* and */. Use blank lines and spaces to separate entries and improve readability. Listing 4-1 illustrates several NETNAMES.DAT entries.

```
NAME: [node or socket name]
ADDRESS: [physical network address]
TYPE: [NODE, LOCAL SOCKET, or REMOTE SOCKET]
OEM: [optional--network vendor defined]
/* comments */
```

Figure 4-2. NETNAMES.DAT Entry Format

The items within each entry are defined as follows:

- **NAME:** Socket and node names can be fifteen characters or less and must consist of letters and numbers only. Do not use punctuation or special characters. The first character in the name must be a letter.

- **ADDRESS:** Node or socket addresses can be 16 characters or less. You have two options for specifying the address:

  - character string: To express the address as a character string, enclose the entry in double quotes; for example, "Node6" or "005".

  - numeric string: To express the address as a numeric string, enter the hexadecimal value for each ASCII character in the address. For example, the ASCII character string address "005" expressed as a hexadecimal numeric string is 30,30,35; "Node6" is 4E,6F,64,65,36. To put a NULL (ASCII 00) in the address, enter two commas one right after the other. See the FlexOS User's Guide for the ASCII to hexadecimal conversion table. The comma does not count in the character total.

  Remote sockets present on multiple nodes must have the same address.

- **TYPE:** The type for each entry must be one of the following:

  NODE - Entry is a node.
  LOCAL SOCKET - Entry is a socket on this node.
  REMOTE SOCKET - Entry is a socket on another node.

- **OEM:** See your network vendor for the description of this field.

Listing 4-1. Sample NETNAMES.DAT Entries

```
/* Sockets                                    */

NAME:          LMAIL
ADDRESS:       "001"
TYPE:          LOCAL SOCKET
/* Local socket for email program             */

NAME:          RMAIL
ADDRESS:       "001"
TYPE:          REMOTE SOCKET
/* Remote socket for email program            */

/* Nodes                                      */

NAME:          HANK
ADDRESS        31,31
TYPE:          NODE
/* This is network address 11                 */

NAME:          MARY
ADDRESS        31,41
TYPE:          NODE
/* This is network address 1A                 */
```

Before proceeding with the next section, make a NETNAMES.DAT file for each node. Use the NAMES PARSE command to check the file for syntax errors. The command form is:

```
A>NAMES PARSE NETNAMES.DAT
```

The NAMES PARSE command tells you which file lines have errors. Record each node's NETNAMES.DAT file on its system: drive and run FSET to give it the -S (system) attribute. FSET is described in the FlexOS User's Guide.

INSTALL ACCOUNTS

Users cannot log on servers that have protection installed unless they have an account there. (Protection is installed on the node with the DEFINE PROTECT=ON command in the node's CONFIG.BAT file.) Accounts are recorded in the node's USER.TAB file.

Create and update the USER.TAB files with a text editor or word processor. Be sure the word processor does not put any special characters in the file. The USER.TAB file must be recorded on the node's system: drive.

Each entry in USER.TAB consists of eight items, separated by commas, arranged in the following sequence:

name,password,userid,groupid,home,wmanager,shell,access

End each entry with a Carriage Return.

See the FlexOS User's Guide for the description of the first seven fields. The eighth field, "access," is a numeric value which states whether the user can log on this node as a local user only, as a remote user only, or as either a local or remote user. The values for the access field are as follows:

Value	Definition
1	Local logon only
2	Remote logon only
3	Both local and remote logon allowed

Take some time to assign each account's user and group IDs. See **Loading Applications and Their Data Files** below for a description of user and group IDs and a strategy for assigning values.

Be sure to enter an account for yourself and to make the user and group IDs zero in this account.

After the accounts have been entered, run the PASSWORD command for each account to set its password. Note that the password cannot be set from a remote node.

SET NETWORK PARAMETERS

FlexNet has a set of parameters that allow you to adjust variable network characteristics. Use the NETSET command to display the default parameter values and modify them for your network. When you modify a NETSET parameter, its new value is not used until the network is restarted. Parameters not changed keep their default value.

Use the following command to show the default parameter values:

A>**NETSET LIST**

To display a brief description of the parameters, enter

A>**NETSET HELP**

Network Parameter Descriptions

Most of the network parameter values shown in the NETSET LIST command should remain as is. These values are set by your vendor to establish essential operating characteristics. However, there are several parameters you can set without jeopardizing network operation. Table 4-1 lists the NETSET parameters you can change and explains their purpose.

See Section 5, "FlexNet Configuration Options," of the FlexNet OEM and Programmer Guide for more detailed information about the network parameters.

Table 4-1. NETSET Parameters

Parameter	Description
NSESS	**Maximum number of connections:** A connection is required for each logon. On requester/server nodes, this includes logon requests issued by the requester mechanism and the requests accepted by the server side. In addition, network applications might also require their own connections to send and receive messages. Refer to the program's manual to determine how many connections you should allow for network applications. Each connection requires 256 bytes of memory in the FlexNet internal memory pool (see NIMP below). If you change NSESS, adjust NIMP accordingly.
NLOGR	**Maximum number of requester logons:** Requester logons are connections to requester/servers established by the user. This value determines what number of the total connections (NSESS) can be used to log on remote servers. The limit is applied on a user window basis. For example, a value of 3 lets each user on the computer log on three servers from each window. A zero value imposes no limit.
NLOGS	**Maximum number of remote log ons:** Remote logons are connections to this node established by remote requesters. This value determines what number of the total connections (NSESS) are allocated to remote requesters. A zero value imposes no limit.

Table 4-1. (Continued)

Parameter	Description
NNTE	**Maximum number of memory-based name translations:** Memory-based name translations are entries read from the NETNAMES.DAT file and stored in memory. Putting the entry in memory provides faster translation.
EXTSEC	**Server Security Mode:** Set this value to 1 to select extended security or to 0 for normal security.
NIMP	**Size of FlexNet internal memory pool:** The FlexNet internal memory pool is the storage area for the data used to manage connections. Increase this value by 256 for each NSESS connection you add. Do not decrease this value without consulting your network vendor.
PKEY	**Password encryption key:** FlexNet uses this value to modify its password scrambling mechanism. Nodes can have different PKEY values, but requesters can log on only servers that have the same key. You can express the encryption key as a hexadecimal value or a character string. To express it as a hexadecimal value, precede the value with 0x. For example, to use the number 88, enter 0x88. Hexadecimal keys must have an even number of digits. To express the key as a character string, enclose the characters in quotation marks. For example, to use the string XXX, enter "XXX".

Changing Network Parameters

There are two ways to change network parameters:

- Change the parameters directly with NETSET commands.
- Create a parameter resource file.

Both means use the NETSET command. In the first option, you specify the parameter by name in the NETSET command line. In the second option, you specify a file in the NETSET command line. The file consists of a sequence of parameter changes. Regardless of the option you select, all changes must be made before you set the node's local name with the NAMES SET LOCAL command.

Note: You should invoke NETSET from within the CONFIG.BAT boot script to minimize the chances of a user error. Invoking NETSET and NAMES is demonstrated in the CONFIG.BAT excerpt in "Update Each Node's CONFIG.BAT File" below.

NETSET Command Form

The NETSET form used to change a parameter directly is

```
NETSET parm=value
```

where "parm" is one of the parameters listed in Table 4-1 above and "value" is its value. For example, to set the PKEY string to AA, enter

```
NETSET PKEY="AA"
```

You can set multiple parameters in the same line by separating each option with a space. For example, the following command sets the PKEY, NSESS, and EXTSEC parameters:

```
NETSET PKEY="AA" NSESS=12 EXTSEC=1
```

If you intend to set any parameters directly with NETSET, make a list of the parameters and their values for each node before proceeding with the next section. Skip ahead to "Update Each Node's CONFIG.BAT File" for instructions on putting the NETSET command in the nodes' CONFIG.BAT file.

Creating the Parameter Resource File

The NETSET parameter resource file consists of a string of statements in the format

```
parm=value
```

where "parm" is one of the parameters and "value" is the new value. Separate each statement with a space or a RETURN. To insert comments, begin the comment with the /* characters and end it with the */ characters. Listing 4-2 illustrates the entry format in a NETSET parameter file.

Listing 4-2. Sample NETSET Parameter File

```
/* Timeout parameters                    */

STMO=5 CTMO=30

/* Node capacity                         */

NSESS=15
NLOGR=3
NLOGS=12
NNTE=3

/* Server Characteristics                */

EXTSEC=0
PKEY=0x5A
```

Edit the resource file with a text editor or word processor. Be careful that the program does not insert any special characters into the file. Make a backup copy of the resource file provided by your network vendor before you edit it.

You use the NETSET FILE command to set the parameters from a resource file. The command form is

```
NETSET FILE=filename
```

Listing 4-3 below includes a demonstration of the NETSET FILE command.

Before proceeding with the next section, create the NETSET parameter resource file for each node. Store the file on the node's system: disk and give it the -S (system) attribute with FSET.

UPDATE EACH NODE'S CONFIG.BAT FILE

FlexNet is composed of a set of drivers--individual program modules provided as files on the FlexNet distribution diskette. Table 4-2 describes the drivers. You do not need to load all of the drivers provided to run FlexNet.

You install the network by invoking the DVRLOAD command from the CONFIG.BAT file to load the drivers required. Besides loading the drivers, you should invoke NETSET and run NAMES SET LOCAL in CONFIG.BAT.

The CONFIG.BAT excerpt in Listing 4-3 loads the FlexNet drivers necessary for a requester/server node, invokes NETSET, sets the LOCAL node name, and enables protection. The DRVLOAD (load driver) commands assume that the system: drive has been previously defined and network driver files are recorded there.

Listing 4-3. CONFIG.BAT Excerpt

```
REM     Install FlexNet drivers.
REM     NETMAN must be first.
REM
DVRLOAD NETMAN: SYSTEM:FLXNETR.DRV PRWSNL
REM
REM     Now install name server driver
REM
DVRLOAD NAMES: SYSTEM:NSD.DRV PRWSNL
REM
REM     Now load the server. This
REM     driver is not necessary
REM     for requester-only nodes.
REM
DVRLOAD SVR: SYSTEM:FLXNETS.DRV PRWSNL
REM
REM     Include the following driver only
REM     if you have network apps. that
REM     require it.
REM
REM     DVRLOAD NET: SYSTEM:NETDEV.DRV PRWSNL
REM
REM     Load the network vendor's
REM     transport driver
REM
DVRLOAD XPORT: SYSTEM:xport.drv PRWSNL
REM
REM     Make changes to network
REM     parameters using PARM.DAT
REM     file. This command must
REM     precede setting local name.
```

```
REM
NETSET FILE=SYSTEM:PARM.DAT
REM
REM     Put any additional changes
REM     necessary here.
REM
REM     Time to set local name; no more
REM     NETSETs allowed.
REM
NAMES SET LOCAL JOE
REM
REM     Enable protection and set
REM     default security word for
REM     node users.
REM
DEFINE -S PROTECT=ON
SECURITY -O=RWED -G=RE -W=RE
REM
REM     Load the log on process
REM
BACK LOGON
REM
REM     End of configuration.
```

Table 4-2. FlexNet Drivers

Driver File Name	Description
FLXNETR.DRV	This driver contains the FlexNet interface to FlexOS and the FlexNet requester mechanism. Load either this driver or FLXNETDR.DRV on each node. Use this driver when the network contains only FlexOS requester/servers.
FLXNETDR.DRV	This driver contains the FlexNet interface to FlexOS and the FlexNet requester mechanism. Load either this driver or FLXNETR.DRV on each node. Use this driver when the network includes PC DOS servers.
NETDEV.DRV	This driver provides applications direct access to the network. Load this driver only if you have a network application that requires it.
NSD.DRV	This driver provides the FlexNet name service. Load this driver on all nodes.
FLXNETS.DRV	This driver provides the FlexNet server mechanism. Load either this driver or FLXNETDS.DRV to make a requester/server node. Use this driver when the requester nodes are all FlexOS requesters or requester/servers.

Table 4-2. (Continued)

Driver File Name	Description
FLXNETDS.DRV	This driver provides the FlexNet server mechanism. Load either this driver or FLXNETS.DRV to make a requester/server node. Use this driver when the requester nodes include PC DOS requesters.
xport.drv	This driver provides the interface to the network hardware. Load this driver on all nodes. The name of this driver is shown in lowercase here and in the sample CONFIG.BAT because this may not be the exact file name. Ask your network vendor for the exact name of the hardware interface driver file for your system.

Before proceeding to the next section, add the following to the CONFIG.BAT file on each node:

- DVRLOAD commands to load the network drivers required by that node
- A NETSET command, if necessary, to modify the default network parameters
- The NAMES SET LOCAL command to set the local node name

If you want to use the FlexOS protection services, you should also set protection and define the default access privileges in CONFIG.BAT.

COPYING FLEXNET FILES

Table 4-3 lists the FlexNet files you need to make FlexOS requesters and requester/servers.

Table 4-3. FlexNet Required Files

File Name	Explanation
Required Drivers	
FLXNETR.DRV or FLXNETDR.DRV	Requester and network control module
NSD.DRV	Name server driver
xport.drv	Network hardware driver
Optional Driver	
NETDEV.DRV	Application interface to network--copy only if required
FLXNETS.DRV or FLXNETDS.DRV	Server module--copy only if the node is a server
Network Commands	
NETSET	Sets network parameters
NAMES	Manages node name directory
NETSTAT	Displays network status
Network Resource Files	
parm.dat	Resource file for NETSET (file can go by a different name)
NETNAMES.DAT	Name service database

Before proceeding with the next section, copy the files required for that node from your backup of the FlexNet distribution diskettes to the node's system: drive. On each node where you have enabled protection, you must also have a USER.TAB file and the LOGON command present on the system: drive.

Once the files have been copied and the CONFIG.BAT file has been modified, FlexNet is installed. Starting FlexOS also loads FlexNet.

LOAD APPLICATIONS

One of your tasks as system manager is to record applications and their data files so that those users who need access have it, while those who should not have access do not. Note that as long as you have an account on the node with user and group IDs of 0,0, you have unrestricted access to all files and directories regardless of the file's or directory's access privileges.

FlexOS limits access to nodes with accounts and passwords. It controls file access through the user and group IDs assigned in the user's account. When a user creates or copies a file, FlexOS records the user's user ID, group ID, and default access privileges with it. When a user trys to use the file, FlexOS compares that user's user and group IDs against the file owner's user and group IDs. Depending on the results of the comparison, the user gets the owner, group, or world access privileges.

FlexOS also records the user's user and group IDs and the default access privileges when he or she creates a subdirectory. As with files, FlexOS compares the user's user and group ID against the owner's to determine whether he or she has owner, group, or world access privileges to the subdirectory.

To limit application and data file access, organize the users into groups according to their program and data needs and assign each group a different number. Set each user's group number in the group ID field in his or her USER.TAB account. Users who require access to different groups of files might require more than one account. Note that users can have different user and group IDs on different nodes.

To complete loading applications, copy the program and data files from the application's distribution diskettes to the network nodes. When you copy the files, be sure you are logged on under an account with the user and group IDs you want permanently recorded with the file. If you log on as the system manager, the user and group IDs recorded with all files and directories you create are zero.

TEST THE NETWORK

This section describes how to test the following aspects of the network:

- Servers--Can each user log on his or her servers and is the proper security mode implemented?
- Access privileges--Does each user have the correct user and group ID numbers?
- Node names--Are the node names correct for each node and does each user have a suitable names directory?

To prepare for testing, start each network node and load FlexOS. No special commands are needed to load FlexNet; it is loaded along with the operating system.

Note: The following procedures test the FlexNet portion of the network. It assumes that the network hardware, intercomputer cables and, if used, network interface boards, are operating

properly. If in testing you determine that the FlexNet portion is prepared correctly but you cannot establish a connection, call your network vendor for assistance. The problem could be in the hardware.

To test the servers, log on a requester node as the local user rather than as the system manager. Once you are in the shell, log on that user's server. If the server has normal security, log on with DIR; if the server has extended security, log on with LOGON. Failure to log on indicates that the user does not have an account on the server.

On all servers with extended security, try logging on with DIR. You should not be successful. If you are, change the EXTSEC value to one on that server with NETSET.

To test access privileges, log on as a user rather than the system manager and invoke the programs that user needs to run to do his or her job. If data files are used, access them and, if required, make changes to make sure the user has read and write privileges. Finally, use CHDIR to confirm that the user has access to all required subdirectories.

In addition, attempt to run commands or to modify files that should not be accessible. Include accessing directories in your test. If you can use them, something is wrong. Make sure node protection is set on in the node's CONFIG.BAT file, the user and group IDs preclude access, and the access privileges are set properly.

To test the node names, display the user's node directory with the NAMES LIST command. Confirm that the names listed have corresponding nodes by comparing them against your node forms and, for servers, logging on. Use network applications if you have them to test the node names too.

This concludes FlexNet testing. If you have network applications, you should test them at this point. Otherwise, the network is ready for use.

End of Section 4

NETWORK ERROR MESSAGE DESCRIPTIONS

This appendix describes the error messages returned by the LOGOFF, LOGON, NAMES, NETSET, and NETSTAT commands. Network failures that occur during the execution of the FlexOS commands do not return special network error messages

LOGOFF ERROR MESSAGES

- Node does not exist

  There is no node for the name entered.

- Not logged on to specified node.

  LOGOFF entered but user was not already logged on.

- Local node name not set

  Log off of a remote node attempted local node name was set.

- An invalid parameter was passed to logoff

  Node specification in command line is invalid.

LOGON ERROR MESSAGES

- Invalid user access

  User name is not in USER.TAB file.

- This user is not permitted to logon to this node

  User name is in USER.TAB but account does not have local access privilege.

- This node does not exist.

  There is no node corresponding to the name entered.

- Cannot connect to this node.

  Node exists but unable to establish a connection with this node.

- Unable to log on to remote node.

  Node exists and connection established but unable to log on.

- Remote node does not support dialect supported by local node.

  Log on of a PC DOS server attempted with FLXNETR.DRV driver installed. (Load FLXNETDR.DRV instead.)

- You are already logged on to this node.

- You must set your local node name before attempting remote log on.

- Remote logon refused.

  There is no account on the remote node or the password entered was wrong.

- You are not permitted to log on to this node

  User name is in USER.TAB but account does not have remote access privilege.

- Log on limit exceeded for this process
- Unable to open USER.TAB on remote node.
- Error while processing USER.TAB on remote node.

  Disk read error occurred while USER.TAB was being read.
- Network software must be installed before attempting remote log ons.

  Network drivers not loaded.
- Invalid node name specified.

  Node name must be alphanumeric and begin with a alphabetic character.

NAMES ERROR MESSAGES

- Too many paramete s supplied.

  Entered too many parameters in the command line than required for the operation.
- Not enough perameters supplied.

  A parameter was left out of the command line.
- Local name already set.

  Local name set for a second time.
- Local name must be SET before a GET or LIST can be issued.

  Attempted to get the local name or the list of nodes before the local node name was set.
- Unrecognized command.

  NAMES command other than SET, GET, LIST, PARSE, or HELP entered.

- WARNING - Name truncated to

  Node name exceeded 15 characters.

- Names must start with a character in the range A .. Z.

  Node name began with a non-alphabetic character.

- Names must be alphanumeric

  Node name contained punctuation mark, special character, or non-printing character.

NAMES PARSE ERROR MESSAGES

- Last NAME ENTRY incomplete

  A NAME: was specified but ADDRESS: and/or TYPE: were not specified before the end-of-file was encountered.

- Excess data found

  Data found following a valid field entry and it was not a comment.

- No node type specified

  No entry found after the TYPE: keyword.

- Local or remote must be specified for a socket.

  SOCKET specified without indicating whether it was LOCAL or REMOTE.

- Illegal address specified.

  Address entered did not begin with quotation mark or did not contain a valid hexadecimal digit.

- No address specified

  No entry found after the ADDRESS: keyword.

- No terminating " specified.

  Close quotation mark left off of character string address.

- Hexadecimal value greater than 255

  A single hexadecimal entry exceeded two digits.

- Address specified is too long

  Address entered exceeded 16 characters.

- No name supplied

  No entry found after the NAME: keyword.

- Name truncated to

  Name entered exceeded 15 characters.

- Duplicate name found

  Cannot use same name twice regardless of type.

NETSET ERROR MESSAGES

- Positive decimal value expected

  Value entered was expressed as a hexadecimal, octal, or alphanumeric value. All numeric values are decimal.

- Value too large for field

  Value entered exceeds size limit for parameter.

- Hexadecimal value expected

  First character in PKEY entry is neither a " or 0x.

- Illegal character or odd number of characters in hexadecimal value

- Null file name specified

  File specification contained a path specification without a file name.

- Null parameter specified

  No value was provided for a parameter specified.

- Cannot change table values after local node is set

  NETSET cannot be run after setting local node name.

- Unable to get Net Parameter Table

  NETSET run before FLXNETR.DRV or FLXNETDR.DRV driver was loaded.

- Too many syntax errors

  There were too many errors in the NETSET parameter file or command line to continue processing.

- Opening input file

  NETSET could not open the parameter file specified.

- Unexpected end of file

  NETSET read an end-of-file character (CTRL-Z) before a parameter value was provided or a comment was closed.

- Delimiter (=) expected

  Delimiter between parameter keyword and value was left out or an invalid delimiter was used.

- Ending quote (") expected

  PKEY closing quotation mark was omitted.

- Null file specification

  NETSET FILE command does not include a file specification.

NETSTAT ERROR MESSAGES

- Please set up local machine name first

  NETSTAT cannot be run until local node name is set.

- Non-numerical value not allowed

  Non-decimal value entered in response to NETSTAT C time interval prompt.

- More than 5 digits? seems impractical

  Value entered in response to NETSTAT C time interval prompt exceeded five digits.

End of Appendix A

NETWORKING WITH PC DOS NODES

FlexNet allows users at PC DOS requesters to access files on FlexOS requester/servers and users at FlexOS nodes to access files on PC DOS servers. This appendix describes how to integrate FlexOS and PC DOS nodes on the same network.

Note: Having FlexOS and PC DOS nodes on the same network is only possible if the network hardware supports it. This description assumes that the PC DOS nodes are running the PC Local Area Network Program (PC LAN).

HOW TO INCLUDE A PC DOS REQUESTER

To have a PC DOS requester access FlexOS requester/servers, install the PC LAN Program on the computer as instructed in the IBM documentation. You must load the FLXNETDS.DRV driver rather than the FLXNETS.DRV driver to make the FlexOS server.

To let a PC DOS requester log on, the FlexOS requester/server must have an account in the USER.TAB file. Add the account as you would for a FlexOS requester with the following exception: You cannot use a logical name in the home directory specification. For example, if you named the node's hard disk drive HD0: when you loaded it with DVRLOAD and then named it C: with DEFINE, the account's home directory specification for this drive must be HD0:.

To log on the FlexOS node, the PC DOS user enters the PC DOS NET USE command in the following form:

```
NET USE \\node\username password
```

The "username" entry is the account name on the node.

The user has access to the drive and directory specified in the account's home entry (in USER.TAB) and all subdirectories below it. If the home directory is itself a subdirectory, the user cannot go above that directory. If the user requires access to more than one drive, make a separate account entry for each drive. The entries must have different user names.

Unlike users at FlexOS requesters, users on PC DOS nodes can log on multiple accounts on FlexOS requester/servers simultaneously. This allows them access to more than one drive at the same time.

HOW TO INCLUDE A PC DOS SERVER

To have a PC DOS server accessed by FlexOS requesters, install the PC LAN Program on the computer as instructed in the IBM documentation. Invoke the NET SHARE command for each device to be accessed from another node. All devices to be accessed by FlexOS requesters must be assigned a shortname. The password is optional.

To access a PC DOS server, you must load the FLXNETDR.DRV driver instead of the FLXNETR.DRV driver on the Concurrent node.

To log on the device, the FlexOS user enters the LOGON command as described in Section 3. Once logged on, the user has access to that directory and its subdirectories. The user cannot move up the directory tree.

FlexOS users can log on PC DOS nodes multiple times so that more than one directory can be used simultaneously.

End of Appendix B

PRINT SPOOLER SYSTEM USE

FlexNet allows FlexOS requester nodes to use print spoolers on PC DOS servers and PC DOS requesters to use spoolers on FlexOS servers. This appendix describes how to use print spoolers through the FlexNet DOS-Requester and DOS-Server (FLXNETDR.DRV and FLXNETDS.DRV) drivers.

See the FlexOS User's Guide for a description of the FlexOS print spooler and associated commands.

The presentation of information in this appendix assumes that the IBM PC Local Area Network Program (PC LAN) has been installed according to the program's documentation.

FLEXOS REQUESTER TO PC LAN SERVER

As shown in the following examples, use the PC LAN NET SHARE command to make the print spooler accessible and assign it a "shortname" of PRN.

```
NET SHARE PRN=PRN:

NET SHARE PRN=LPT1:
```

Note that the trailing colon of the device name is optional.

To access a PC DOS print spooler, the FlexOS requester nodes must have the FLXNETDR.DRV driver installed. Use the DRVLOAD command in the nodes' CONFIG.BAT file; refer to Section 4 for details.

Before sending requests to the PC DOS print spooler, users at FlexOS requester nodes must use the following DEFINE command to associate the PC DOS node with the PRN device name:

DEFINE -S PRN: = node::prn:

node:: is the name of the PC DOS server node. This command can be included in the requester node's CONFIG.BAT file. It allows you to specify PRN as the target device. For example, you can use either of the following commands to print a file:

COPY filename PRN:

TYPE filename > PRN:

If the PC DOS server node shares the print spooler with a password, users at FlexOS nodes should log on to the print spooler first. See Section 3 for a description of the LOGON command. Supply PRN as the user name. If the PC DOS print spooler is shared without password protection, FlexNet performs the default logon.

PC LAN REQUESTER TO FLEXOS SERVER

To allow PC LAN requesters access to the FlexOS print spooler, the FlexOS server must have the FLXNE DS.DRV installed. See Section 4 for information about driver installation.

To allow a PC DOS requester to log on, the FlexOS server node must have an account in the USER.TAB file. The account name must be the one used as "shortname" or "printdevice" in the NET USE command. For example, a user logging on to the FlexOS server from a PC LAN requester would enter the following command:

NET USE \\node\shortname password

where shortname is the account name of the entry in USER.TAB file. The home item in the USER.TAB entry should be PRN:.

- Users at PC DOS requester nodes can send print jobs to a FlexOS server with COPY or TYPE commands. The PrtSc key function is not allowed in PC LAN over network. FlexNet does not support the PC LAN NET PRINT command.

  The FlexOS print spooler is a separate, loadable module (SPLDRV.DRV). The system manager should make sure that it is defined as the PRN device and installed and running before users at PC DOS requesters send print jobs to the FlexOS server. SPLDRV.DRV installation is described in the FlexOS User's Guide.

  If the printer driver, PRINTER.DRV, (not the print spooler) is defined as the PRN device on the FlexOS server, print jobs might become mixed up during simultaneous requests for the PRN device.

End of Appendix C

INDEX

G

H

I

L

M

N

W

X

MAIL
USER'S GUIDE

1082-2008-001

First Edition: October 1986

Foreword

The MAIL program from Digital Research® is an electronic mail system that runs on the FlexOS™ Operating System, versions 1.3 and greater. The MAIL program runs with or without the FlexNET™ Network Operating System, versions 1.3 and greater, with which it is distributed.

This manual assumes the FlexOS operating system is running on your computer system, and that you know how log on to the system and enter a command at the system prompt.

If your system is running FlexNET, you should be familiar with node names.

Contents of This Manual

The MAIL User's Guide contains the following sections:

- Section 1 introduces you to MAIL.
- Section 2 shows you how to use special MAIL keys.
- Section 3 describes MAIL commands you can use.
- Section 4 shows you how to install MAIL.
- Appendix A describes format of MAIL files.

Conventions

In the text of this guide, command names, directory names, and file names are all in upper-case characters for easy recognition.

In Section 3, MAIL Commands, general terms such as filename and message_number, for which you must substitute your own file name or number, are in lower-case characters.

Examples are in a different typeface, similar to your computer display.

- User input, (exactly what you type), is in **bold-face** characters.
- Computer output to your screen is in normal face characters.

Associated Manuals

The following manuals describe software associated with MAIL:

- FlexOS User's Guide
- FlexNET User's Guide

Contents

Tables

Introduction to MAIL

The MAIL program is an electronic mail system you use to send and receive messages on your computer system.

With MAIL, you can do the following:

- Use the HELP command to display help on how to use a command.
- Press the ESCAPE key to make MAIL finish typing the command for you.
- Press the question mark to learn what you can do next.
- Include a file in a message.
- Use a file that contains a list of names for the recipients when sending mail.
- Correct mistakes with the standard FlexOS command line editing keys.
- Use your favorite text editor to compose your message.

To start the MAIL program, type MAIL at the system prompt, then press the ENTER or RETURN key, as shown in the following example.

```
C>MAIL
MAIL Version 01.00  09-SEP-86
For help, type HELP

5 messages in mailbox, 0 new messages
MAIL>
```

MAIL displays its version number and date, a help message, the total number of messages, and the number of new messages in the mailbox. It then displays its own prompt, MAIL>.

Type EXIT to return to the operating system prompt.

After you start the mail program, you can use the commands described in table 1-1.

Note: As a shortcut, MAIL allows you to type only as many letters of a command as needed to distinguish it from the other commands.

Table 1-1. MAIL Commands

Command	Description
ANSWER	answers a message
DELETE	marks a message sequence for deletion
EXIT	exits the MAIL program
FORWARD	forwards a message to another user
GET	gets another user's mailbox file
HELP	displays information about MAIL
LIST	lists the messages in your mailbox
MARK	marks a message as having been read
OUTPUT	puts a message in a file, possibly for printing
PROFILE	allows you to change mail system settings
PUSH	exits to system prompt while saving mail session
READ	reads a new or specified message
SEND	sends a message
UNDELETE	unmarks a message that was marked for deletion
UNMARK	marks a message as not having been read
VERSION	displays the MAIL software version number

Section 2 describes MAIL features in more detail. Section 3 describes each MAIL command and provides examples.

Using MAIL

This section describes the following:

- FlexOS command line editing keys
- MAIL control keys
- Escape Key
- Question Mark
- MAIL message format

2.1 Command Line Editing Keys

You use the standard FlexOS command line editing keys while you are typing your MAIL message. These keys, described in Table 2-1, allow you to make changes to the line on which you are typing.

Table 2-1. FlexOS Line Editing Keys

Key	Description
Backspace	deletes character to left of cursor
CTRL-B	moves cursor to beginning or end of line
CTRL-X	deletes all characters to left of cursor on the current line
Delete	deletes character at cursor
ENTER	tells MAIL you are done entering a command or line of information
Left-arrow	moves cursor left one character
Right-arrow	moves cursor right one character

If you need to make more extensive changes, you can use CTRL-E to invoke a text editor, as described in Section 2.2.

Section 4.4 tells how to specify your favorite text editor. Otherwise, MAIL starts the DR Edix editor.

2.2 MAIL Control Keys

You can use the special key combinations shown in table 2-2 when you use MAIL. Remember to press the CONTROL key, usually labelled CTRL, and hold it down while you press the letter C, E, F, or Z.

Table 2-2. MAIL Control Keys

Key Combination	Description
CTRL-C	This key combination cancels any command or message and returns to MAIL> prompt.
CTRL-E	CTRL-E invokes the DR Edix text editor when used with the ANSWER, FORWARD, and SEND commands. See Section 4.4 for instructions to define your own text editor. When you exit the editor you return to MAIL.
CTRL-F	CTRL-F prompts you for a filename to include. You can use this to name a list of users to receive your message. You can also name a file that contains the message. See Section 3.1 for a quick review of file specifications.
CTRL-Z	CTRL-Z tells MAIL you are finished typing your message. You must use CTRL-Z at the end of every message.

2.3 Escape Key

You can press the ESCAPE key (usually labelled Esc) at any time while you are entering a MAIL command. MAIL finishes typing the command for you and displays a message telling what the command does.

If you press the Escape key more than once, it displays a square bracket. Delete the bracket and proceed.

When two or more commands start with the same letters, you must type enough letters to identify the command you want. Otherwise, when you press ESCAPE, MAIL displays a message stating that you specified an ambiguous command name.

2.4 Question Mark

You can type a question mark (?) at any time. MAIL displays helpful information about what you should do next. When there is no more information, you see the message: "No more help is available."

When you type part or all of a command followed immediately by a question mark, MAIL displays the names of all the commands that begin with the same letters. Sometimes there is just one.

When you type a command followed by a space then a question mark, MAIL displays helpful information about the command.

2.5 Format of a MAIL Message

When you use the READ command to read a message, MAIL displays information at the top of the message. The information is similar to what is in the following example. The example shows all the information you are likely to see at the top of a message. However, your messages might not contain all of this information.

Format of MAIL Message

```
***** Message #2: 153 characters
Date: 20-October-1986 11:35:49
Message-id: 1986-09-20-11:35:49,875+16:00/F&A::Beth
From:  F&A::Beth
To: ENG::Tim
Cc: Steve
Subject: Re: Monday morning meeting
In-reply-to: 1986-09-19-10:30:20,343+16:00/ENG::Tim
```

Each line has the following meaning:

- The top line contains five asterisks, the message number, and the number of characters in the message.
- The "Date:" line contains the date and time (hours, minutes, and seconds) the message was sent.
- The "Message-id:" line contains the message identification code. This code is for the system and you do not need to read it.
- The "From:" line tells you who the message is from.
- The "To:" line tells you who the message is to.
- The "Cc:" line, if present, contains the name of the user or users to receive a courtesy copy of the message.
- The "Subject:" line, if present, tells you the subject of the message.
- The "In-reply-to:" line, if present, contains the message-id of the message being answered.

The message text always ends with this symbol: <*>.

2.5.1 User Name

The user name you see after "To:," "From:," and "Cc:" in the mail message can appear in several forms, as shown in Table 2-3.

The LIST command displays names in the same forms.

Table 2-3. Forms of User Names

Form	Description
Bruce	This is a name of a user on the same node as the sender or receiver. A simple name like this can also originate from the Mail Forwarding Data Base file described in Section 4.2. This file allows you to define a name to use instead of typing a complete node and username when you send a message.
FINANCE::BETH	This is a standard DR Net node name and user name. This message came from or is going to another network node.
Pooh Bear<Jo>	You can also set a personal name with the PROFILE command. The personal name only appears in the "From:" line. Personal names are always followed by the real user name enclosed in angle brackets. You need to use the real name when you are sending mail to a user.
Mail Delivery Process	This is a message from MAIL. You receive such a message when MAIL cannot deliver your mail message to a node or user for some reason.

End of Section 2

MAIL Commands

This section includes:

- a brief review of the file specification
- instructions to send a file directly from the command line
- instructions to specify a message sequence
- a description of each MAIL command

3.1 File Specification

Recall that a file specification tells the operating system the location of a file. A complete file specification takes the following format:

```
Node::Drive:/Directory_path/Filename.Ext
```

You usually need to specify only the filename, period, and filename extension. If you are on a network system and you want to send a file to another node, you must specify the node name.

3.2 Command Line

You can send a file from the command line without using the SEND command. Use the following form of the MAIL command:

```
C>MAIL file_specification user_list "subject"
```

The file_specification, user_list, and subject make up the command tail. The command tail can contain 128 characters.

The user_list can contain the names of one or more users, separated by commas. You cannot have a space between user names in the user_list.

Put quotes around the subject, which can contain up to 80 characters.

The following command line sends a file on drive B called MEMO.TXT to Pat, Bruce, and Steve. MAIL inserts "Company Picnic" as the subject of the message.

```
C>MAIL B:MEMO.TXT PAT,BRUCE,STEVE, "COMPANY PICNIC"
```

3.3 Message Sequence

Several MAIL commands allow you to specify a group of messages. Such a group is called a message sequence.

Table 3-1 shows the ways you can specify a message sequence. The LIST command displays message numbers and user names of messages currently in your mailbox.

Note: In Table 3-1, CAPITAL letters indicate keywords that you type exactly as shown. Lower-case words such as number and username require you to substitute the appropriate message number and user name.

Table 3-1. Forms of a Message Sequence

Form	Description
Number	This is the number of a single message.
ALL	All indicates every message in the mailbox.
Number-Number	This indicates a range of messages, beginning with the first number and including the last. You can separate the numbers with a dash (-) or a colon (:).
FROM Username	This command returns all messages from specified user. You can specify as little as one character for the username. MAIL displays all messages from users that have that character or characters in the username.
DELETED	DELETED returns all messages marked with a D for deletion.
UNDELETED	UNDELETED returns all messages not marked with a D for deletion.
NEW	NEW gives you all messages that have arrived since you exited MAIL
READ	All messages that have been read with the READ command, put in a file with the OUTPUT command, or marked as having been read with the MARK command are listed when you enter READ.
UNREAD	All messages that have not been read with the READ command, put in a file with the OUTPUT command, or marked as having been read with the MARK command are listed when you enter UNREAD.
Combinations	You can enter any combination of the above, separated by commas.

ANSWER

Forms: ANSWER
ANSWER message_number

Explanation: Use the ANSWER command to respond to a message without having to type the addressee or the subject. MAIL automatically supplies the name of the sender and the subject of the message you are answering. To reply to a message other than the last message you read, enter the message number you want to respond to.

Examples:

```
MAIL>ANSWER
```

ANSWER, by itself, answers the last message you read.

```
MAIL>ANSWER 5
```

This example answers message number 5. Use the LIST command to display message numbers.

DELETE

Forms: DELETE
DELETE message_sequence

Explanation: Use DELETE to mark a message for deletion. DELETE places a D in column 3 of the LIST display. The message is not actually deleted until you exit MAIL. You can use UNDELETE to unmark the message if you change your mind.

Delete messages to reduce the amount of disk space used by the mailbox file.

Examples:

```
MAIL>DELETE
```

DELETE, by itself, deletes the last message you read.

```
MAIL>DELETE 13
```

This example deletes message number 13.

```
MAIL>DELETE 13-15
```

This command deletes messages 13 through 15.

```
MAIL>DELETE FROM HARRY
```

This command deletes all messages from the user named Harry.

EXIT

Form: EXIT

Explanation: Use EXIT to exit MAIL and return to the operating system prompt.

MAIL asks if you want to delete any messages marked for deletion before exiting. You can type Yes, No, Y or N. See the PROFILE command to set MAIL so it does not ask this question.

Example:

```
MAIL>EXIT
```

EXIT leaves MAIL and returns you to the operating system prompt.

FORWARD

Forms: FORWARD
FORWARD message_number

Explanation: Use FORWARD to send a copy of a message to another user. MAIL automatically uses the message number of the last message you read. If you did not read a message, or you want to forward a different message, enter the message number.

Notice that MAIL displays the heading of the message you are forwarding.

You are prompted for the forwarding name and subject of the message. If the Profile file is set to Cc:, you are prompted for names of users to receive a courtesy copy of the forwarded message. See the PROFILE command in this section.

You can add your own comments to the end of the original message by typing them after the prompt, then typing Ctrl-Z to end the message.

Examples:

```
MAIL>FORWARD
```

FORWARD, entered alone, begins the prompting sequence to forward the last message you read.

```
MAIL>FORWARD 23
```

FORWARD, entered with the number 23, begins the prompting sequence to forward message number 23.

GET

Forms: GET
GET file_specification

Explanation: Use GET to change the mailbox file that MAIL is currently using. You must have access privileges to the mailbox you want to "get."

GET displays the number of messages in the named mailbox file.

If you do not specify a filename, GET returns you to your own mailbox.

GET remains in effect until you issue another GET command or exit MAIL.

Mailboxes are located in a directory called SYSTEM:MAIL and have a filename extension of MBX. See Section A.1, Mailbox Files, for more information.

Examples:

```
MAIL>GET SYSTEM:MAIL\STEVE.MBX
```

This example gets the mailbox STEVE.MBX from the SYSTEM:MAIL directory.

```
MAIL>GET
```

GET, by itself, returns you to your own mailbox if you were using another, and displays the number of messages in it.

```
MAIL>GET SYSTEM:MAIL\BULLETIN.MBX
```

In this example, GET retrieves the mailbox BULLETIN.MBX from the SYSTEM:MAIL directory.

HELP

Forms:

HELP
HELP command_name
HELP topic

Explanation:

Use HELP to display information about MAIL commands. The display shows the command name, what you can type after the command name (the argument), and a description of the command's function. HELP also displays topics on which you can get information.

HELP with the command_name displays information on that particular command.

HELP with a topic displays information on that topic.

You need to type only enough letters to uniquely identify the command or topic.

Examples:

```
MAIL>HELP
```

HELP, by itself, displays MAIL commands and topics.

```
MAIL>HELP SEND
```

HELP with SEND displays information about the SEND command.

```
MAIL>HELP ADDITIONAL_COMMANDS
```

This example displays a list of less frequently used commands.

```
MAIL>H ARG
```

This command line contains enough letters to tell MAIL that you want HELP (H) on the topic ARGUMENTS (ARG).

LIST

Forms:

LIST
LIST message_sequence

Explanation:

Use LIST to display information about your mail messages.

See Section 2.5.1, User Name, for an explanation of the forms in which the sender's name can appear. If the sender's name or the message subject is longer than the column-width allows, LIST displays only part of it.

You might see Re: before some of the message subjects. When you use the ANSWER command to send a message, DR MAIL automatically puts Re: in front of the original subject.

The numbers in the following list correspond to the columns in the LIST display, starting from the left. The letters in parentheses correspond to letters you might see in columns 1, 2, and 3. Note that the first three columns might be blank.

The LIST display tells you the following information:

1. The message is new (N) or not new (blank).
2. The message has been read (blank) or not read (U).
3. The message is marked for deletion (D) or not marked for deletion (blank).
4. Column 4 shows the number of the message.
5. Column 5 shows the name of the sender.
6. Column 6 shows the subject of the message.
7. Column 7 shows the number of characters in the message.

Examples:

```
MAIL>LIST
```

LIST, by itself, displays information for all messages in the current mailbox.

```
MAIL>LIST FROM SA
```

This command displays information for all messages in the current mailbox from any user with the letter sequence SA anywhere in the user name.

```
MAIL>LIST 3-8
```

This example displays information for messages numbered 3 through 8 in the current mailbox.

```
MAIL>LIST NEW
```

This command displays information for all new messages in the current mailbox.

```
MAIL>LIST DELETED
```

This example displays information for all messages marked for deletion.

MARK

Form: MARK message_sequence

Explanation: When you read a message, MAIL automatically marks the message as having been read. The MARK command marks a message as having been read even if you have not read it.

Use UNMARK to mark the message as not having been read.

The LIST command displays a blank in column 2 for messages that have been read.

Example:

```
MAIL>MARK 1-5
```

This example marks messages 1 through 5 as having been read.

OUTPUT

Forms: OUTPUT
OUTPUT message_sequence

Explanation: Use OUTPUT to save a copy of a message in a file. You can then print the file if you wish. If you do not specify a message number or message sequence, MAIL automatically files the last message you read. MAIL prompts you for the filename. If the file does not exist, MAIL creates a new file. If the file does exist, the message is appended to it.

Examples:

```
MAIL>OUTPUT
Filename:
```

OUTPUT, by itself, displays the filename prompt and writes the last read message to the file you specify.

```
MAIL>OUTPUT 1-5
Filename:JULY.MSG
```

This example puts messages numbered 1 through 5 in a file called JULY.MSG in the current directory, or adds them to an existing file called JULY.MSG in the current directory.

```
MAIL>OUTPUT ALL
Filename: C:MAGGIE\KEEP\MAIL.OLD
```

All messages are put in a file called MAIL.OLD in the directory MAGGIE\KEEP on Drive C.

```
MAIL>OUTPUT FROM HARRY
Filename:  ACCOUNTING::PROJECT\MEMOS.JUN
```

All messages from user HARRY are put into the file MEMOS.JUN on the node ACCOUNTING:: in the directory PROJECT.

PROFILE

Form: PROFILE

Explanation: Use PROFILE to change the MAIL default settings. See Section A.3, Profile File, for more information.

PROFILE displays ten questions. The previous settings are shown in parentheses. Press RETURN to leave them the same. You can type Y or N to answer the yes/no questions.

Examples:

```
Mail system prompt (MAIL>):
```

You can enter up to 79 characters to be the new mail system prompt.

```
Personal name (no default):
```

You can enter any name up to 79 characters. The personal name appears after From: in the messages you send. You cannot use the left-angle, right angle, or comma (<>,) characters in a personal name.

```
List new messages on startup (YES):
```

YES tells MAIL to list new messages when you start the system.

```
Do --MORE-- on READ (YES):
```

YES tells MAIL to pause after each screen of text, until you press a key to continue reading your messages.

Include self in reply on ANSWER (NO):

NO tells MAIL not to send a copy of your replies to yourself when you use ANSWER.

Always send copy of message to self (NO)

NO tells MAIL not to send a copy of a message to yourself when you use ANSWER, FORWARD, or SEND. You can name yourself in the To: or Cc: list to send yourself a particular message.

Ask for Cc list in SEND/FORWARD (YES):

YES tells MAIL to ask for Courtesy Copy list for each message you send.

Ask for Subject in SEND/FORWARD (YES):

YES tells MAIL to ask for a Subject for each message you send.

Formfeed between messages in OUTPUT (YES)

YES tells MAIL to use a form feed to separate messages you have OUTPUT to a file. NO causes a line feed to separate each message.

Ask about purging deleted messages on EXIT (YES):

YES tells MAIL to ask before deleting messages when you issue an EXIT command. NO causes messages to be deleted automatically when you EXIT.

PUSH

Form: PUSH

Explanation: PUSH saves the current mail session and starts the operating system program from which MAIL was invoked. To return to DR MAIL, type EXIT next to the operating system prompt.

Examples:

```
MAIL>PUSH
Running SHELL.
Type EXIT to return to MAIL.
C>DIR
.
.
.
C>EXIT
MAIL>
```

In this example, PUSH has invoked the operating system user interface program, which is referred to as the Shell. The Shell displays the C> prompt, indicating that the default drive is Drive C.

The user then typed the DIR (directory) command. The dots indicate the directory display for Drive C. When the directory display ends, the operating system displays the C> prompt again. The user types EXIT and returns to the MAIL program, which displays the MAIL> prompt.

READ

Forms: READ
READ message_number

Explanation: Use READ to read your mail messages. READ with no number displays the first newly received message, or the message whose number follows the most recently read message.

Examples:

```
MAIL>READ
```

The commane READ, by itself, displays a new message, or the message following the last message you read.

```
MAIL>READ 20
```

This example displays message number 20.

SEND

Forms: SEND

Explanation: Use SEND to send a message to another user or list of users. Ask your system manager for a list of valid user names. Section 4.3, USER.TAB File, contains more information on user names.

You can type CTRL-F to include a file containing valid user names. DR MAIL sends the message to every name in the file.

Examples:

```
MAIL>SEND
To: Personnel::Susan
Cc: Joe, Steve, Elaine
Subject: Company Picnic
Enter message text, ending with Ctrl-Z.
Ctrl-C cancels, Ctrl-E runs editor
Ctrl-F includes file.
----------
Hi Susan,

Are You going to the picnic this
Saturday?  Do you want to carpool?
Please let me know.

Thanks,  Louisa
```

This example sends Louisa's message about the company picnic to node PERSONNEL, user SUSAN, with courtesy copies to JOE, STEVE, and ELAINE.

```
MAIL>SEND
To: (CTRL-F) Filename: ACOUNTNG.LIS
Subject: PAYROLL
Enter message text, ending with Ctrl-Z.
Ctrl-C cancels, Ctrl-E runs editor
Ctrl-F includes file.
----------
Please advise me if you are not
on schedule.
```

After the To: prompt, the sender typed Ctrl-F. MAIL then prompted for the name of a file to include. The sender entered the filename ACOUNTNG.LIS. The file should contain the names of users with mailbox privileges. MAIL looks for that file in the current directory. If it does not find the file, MAIL displays an error message.

Note that the filename can be a full file specification containing a node name, a device name, a directory path, a filename, and a filename extension.

UNDELETE

Forms: UNDELETE message_sequence

Explanation: Use UNDELETE to restore a message you have marked for deletion with the DELETE command. You must use UNDELETE before you exit DR MAIL. UNDELETE places a blank in the third column of the LIST display.

Examples:

```
MAIL>UNDELETE 3-5
```

This example restores message numbers 3 through 5, and removes the D (for DELETE) from column 3 of the LIST display.

UNMARK

Form: UNMARK message_number

Explanation: Use the UNMARK command to mark a message as not having been read. MAIL places a U in the second column of the LIST display.

Examples:

```
MAIL>UNMARK 3-5
```

Messages 3 through 5 are marked as not having been read.

VERSION

Form: VERSION

Explanation: Displays the version number of the MAIL software on your system.

Example:

```
MAIL>VERSION
MAIL Version 01.00 09-SEP-86
```

End of Section 3

Installing MAIL

MAIL requires a computer system running the FlexOS operating system. MAIL is distributed with FlexNet 1.3, although it runs on FlexOS without the network.

The FlexNet distribution disks contain the files MAIL.286 and MDP.286.

You do not need to create any files to run MAIL. However, as System Manager, you should know about the following files:

Table 4-1. MAIL Files

Filename	Description
MAIL.286	user interface program to copy into the users' search path
MDP.286	mail Delivery Process program to copy into system search path
USER.TAB	existing FlexOS system file that by default gives all users (identified by USERNAME) the privilege of having a mailbox
USERNAME.MBX	mailbox file created by MAIL when a user first receives mail
MFDB.TXT	optional Mail Forwarding Data Base file that you can create
USERNAME.PRF	profile file created by MAIL when the user runs the PROFILE command and changes the default options

Follow the steps below to install MAIL:

1. Create the following two directories on the node that is to run MAIL:

   SYSTEM:MAIL — This directory holds the user mailbox files (USERNAME.MBX). Also holds the Mail Forwarding Database file (MFDB.TXT) and the profile file (USERNAME.PRF) if they exist.

   SYSTEM:MAIL/OUTGOING — This directory holds messages that are waiting to be forwarded to other nodes.

2. Copy MAIL.286 into the system directory or into a directory that is in the users' search path so the user can run it by simply typing MAIL at the system prompt.

3. Copy MDP.286 into a directory that is in the system search path so FlexOS can find it.

4. Enter the following command line in the CONFIG.BAT file. If you are installing a network, place this command after the command that sets the local node name:

   ```
   BACK MDP
   ```

5. Create the username MAILER in the USER.TAB file on every node that is to run MAIL.

   See Sections 4.1, Mail Delivery Process, and 4.3, USER.TAB File, for more information on this file.

   The following example shows a reasonable entry for user MAILER in a USER.TAB file:

   ```
   MAILER,********,50,3,,,,6
   ```

   Table 4-2 describes each field in the USER.TAB file for the user MAILER.

Table 4-2. Fields in USER.TAB File

Field	Entry and Description
Username	MAILER. This username is necessary to send mail to another node.
Password	********. This field must contain eight characters. It should contain a password to prevent an unauthorized person from logging on under the user name MAILER. The usual method of creating a password is to enter 8 asterisks into this field in the USER.TAB file, and then use the PASSWORD command to set a password. The password then appears in the USER.TAB file in encrypted form.
User-id	50. This must be a unique user-id.
Group-id	3. This can be any group-id.
Home	MAILER is for remote logon only, so the concept of a home directory on a local node is irrelevant. This field can be empty.
Wmanager	This field is for local logon only and can be empty. FlexOS supplies a default value for this field.
Shell	This field pertains to local logon and can be empty. FlexOS supplies a default value.
Access	6. A value of 6 in the access field means the user can only log on from a remote node, and cannot have a mailbox. MAILER does not need a mailbox because it is not a real user. If this number were 3, and no password was required, an unauthorized user could log on to a local node as MAILER.

MAIL creates a mailbox file (USERNAME.MBX) as needed. It creates a profile file (USERNAME.PRF) when a user runs the PROFILE command in MAIL.286 and changes the default settings. MAIL places both files in the SYSTEM:MAIL directory.

You might want to create a Mail Forwarding Data Base file (MFDB.TXT). See Section 4.2 for more information.

4.1 Mail Delivery Process

The Mail Delivery Process, MDP.286, normally runs in the background. It produces no console output and runs until canceled.

MDP.286 must be running to send a message. If it is not running when MAIL.286 is invoked, the user can still use the MAIL commands that do not send a message.

When MDP.286 starts, it attempts to set its local LOGON name in the LOGON table to MAILER. Having a user name allows MDP.286 to logon to another node having the same user name without entering a password. This is necessary to deliver mail to another node.

MDP.286 can successfully set its logon name to MAILER if it is started by the CONFIG.BAT file, or by a user with a user-id and group-id of 0,0 (System Manager privileges).

If a user without system privileges starts MDP.286, then it will run under that USERNAME, and can only access another node if the same USERNAME has an account on that node.

Therefore, every node that is to run MAIL should have a user MAILER in the USER.TAB file for that node.

The logon name should affect only the MDP.286 process and not other system processes, so it should have its own family-id. The BACK command does this automatically.

For more information on logging on and the USER.TAB file, see Appendix A, "System Manager," in the FlexOS User's Guide, and "Logging On" in Section 2 of the FlexNet User's Guide.

The Mail Delivery Process cannot send mail to an extended security node at this time. An extended security node requires a user to explicitly type in a password and MDP.286 cannot do this.

4.1.1 Log File

MDP.286 can create a log file in a specified directory or in the current working directory when MDP.286 is started. If you want a log file, specify a filename in the command tail when you start MDP.286.

The following command line produces a log file called MONDAY.LOG in the directory LOGFILES on Drive B.

```
C>MDP B:LOGFILES/MONDAY.LOG
```

The following command line produces a log file called MONDAY.LOG in whatever is the current working directory on Drive C.

```
C>MDP MONDAY.LOG
```

4.2 Mail Forwarding Data Base File

You might want to create a Mail Forwarding Data Base file to:

- change the name under which you receive mail
- send mail to a list of users by typing one user name
- simplify names to type when sending a message to another node
- forward mail automatically to another user or another node

You must name this file MFDB.TXT and place it in the SYSTEM:MAIL directory.

The file must contain:

- a name (alias) followed by a colon
- one or more mailbox names, separated by commas

A mailbox name consists of a node and a username, or, in the case of a mailbox on a local node, just a user name.

A message addressed to the alias is delivered to all the users in the user list.

The following example shows a sample MFDB.TXT file.

```
Tim: Accounting::Payroll
Elaine: Personnel::Elaine
Testers: Tim, Elaine, Steve, Peter, Rob
```

The aliases are:

- Tim
- Elaine
- Testers

The mailbox names are:

- Accounting::Payroll
- Personnel::Elaine
- Steve
- Peter
- Rob

Notice that you can use an alias in the user list, as in the line Testers: in the previous example.

Blank space is allowed after the colon in the alias name, and after commas in the mailbox list. Any combination of spaces, tabs, and carriage return/linefeed constitute blank space.

4.3 USER.TAB File

The USER.TAB file is an ASCII file that exists in the FlexOS operating environment. For more information see Appendix A, "System Manager," in the FlexOS User's Guide, and Section 4, "Network Installation," in the FlexNet User's Guide.

By default, the USER.TAB file is set up to allow a user to have a mailbox. You can change the USER.TAB file with a text editor, but you do not need to unless you want to specify that a particular user cannot have a mailbox.

The USER.TAB file is in the SYSTEM: directory on each network node. It contains eight fields separated by commas. The eighth field is the access field. It is a numeric value indicating whether the user can log onto the node as a local user, a remote user, or both. It also indicates whether the user is allowed to have a mailbox.

You can remove the mailbox privilege by changing the access field. Table 4-3 shows each valid value in the access field and its meaning.

Table 4-3. Values of Access Field in USER.TAB File

Value	Definition
0	no logon privilege
1	local logon only; user has mailbox privilege
2	remote logon only; user has mailbox privilege
3	local and remote logon; user has mailbox privilege
4	no logon privilege; no mailbox privilege
5	local logon only; no mailbox privilege
6	remote logon only; no mailbox privilege
7	local and remote logon; no mailbox privilege

If the value of the access field is 0-3, the user can have a mailbox. The Mail Delivery Process (MDP.286) delivers mail to the mailbox file USERNAME.MBX if it exists. If the mailbox file does not exist and the user receives a message, MDP.286 creates the mailbox file and delivers the message.

If the value of the access field is 4-7, the user cannot have a mailbox on the node. The Mail Delivery Process will not deliver mail to him even if a mailbox file with the proper name exists.

4.4 Defining the Text Editor

Each user can specify a text editor to use with MAIL.

Use the FlexOS DEFINE command to specify the text editor you want to use when you press CTRL-E. MAIL looks for the keyword "editor" in the definition table created by the DEFINE command.

If you do not assign an editor, MAIL looks for DREDIX.286 in the system search path.

You can specify a path in the DEFINE command.

The following command line defines the text editing program named TEXTSTAR.286 as the user's choice. TEXTSTAR.286 must be in the SYSTEM: directory.

```
C>DEFINE Editor = SYSTEM:/TEXTSTAR.286
```

End of Section 4

MAIL File Formats

This section contains formats for the following record headers, files, and messages:

- Mailbox File Record Header
- Mailbox File
- Profile File
- Pipe Message
- Queued Message File
- Mail Message

A.1 Mailbox File Record Header Format

A mailbox file (USERNAME.MBX) consists of zero or more variable-length records. Each record starts with a 34 byte fixed-length header that indicates the length of the message and some attributes.

The mailbox file record header is in the following format:

*****nnnnnnnnnn<sp>bbbbbbbbbbbbbbb<cr><lf>

Table A-1 describes each field in the record header.

Table A-1. Format of Mailbox File Record Header

Field	Description
*****	The five asterisks are the first field in every header. If they are not there, the file is corrupted.
nnnnnnnnnn	The n's indicate the length of the current field in bytes, excluding record header bytes. Each n is a decimal digit. Leading zeroes can be replaced by spaces.
<sp>	This is the ASCII space character.
bbbbbbbbbbbbbbbb	The b's represent 16 message attribute bits. Each b is 0 or 1; leading zeroes are NOT suppressed. The low-order 3 bits are defined below; the remaining bits are RESERVED. 001 If 1, this message has not yet been read (UNREAD). If 0, this message has been read. 010 If 1, this message arrived after the last time the MAIL program was exited (NEW). 100 If 1, the message has been marked for deletion (DELETED). It should be removed from the file when the user issues an EXIT command. 011 If both bits are set to 1, this is a newly arrived message (UNREAD/NEW).
<cr>	This is the ASCII Carriage Return character.
<lf>	This is the ASCII Line Feed character.

A.2 Mailbox File Record Format

A mailbox file record contains the message. The message consists of message header information and the body or actual text of the message. The header information is encoded as follows:

<field num>: <field value><cr><lf>

Message header fields can appear in any order, except for the field number -3:, which must always be the last field in the message header. This field marks the end of the message header and the beginning of the message text.

Note that these encodings are based on the standards recommended in the International Telegraph and Telephone Consultative Committee (CCITT) 1985 Red Book, Fascicle VIII.7, "Data Communication Networks Message Handling Systems," Recommendations X.400 - X.430.

Table A-2 shows valid values for <field num>.

Table A-2. Field Number and Description of MAIL Message

Field Number	Description and Field Value
-3:	Message text begins here. <field value> is null. The text begins on the next line and continues until the next record header. This is always the last field in the message header.
-2:	IPMessageID. (**Message-id:** field). <field value> is a text string.
-1:	Send Date. (**Date:** field). <field value> is a UTC (Universal Time Coordinate) date string.
0:	Originator. (**From:** field). <field value> is a mailbox name.
1:	AuthorizingUsers. (Not implemented). <field value> is a list of mailbox names.
2:	PrimaryRecipients. (**To:** field). <field value> is a list of mailbox names.
3:	CopyRecipients. (**Cc:** field). <field value> is a list of mailbox names.
4:	BlindCopyRecipients. (Not implemented). <field value> is a list of mailbox names.
5:	InReplyTo. (**In-reply-to:** field). <field value> is a message id (text string).
6:	Obsoletes. (Not implemented). <field value> is a list of message ids.
7:	CrossReferences. (Not implemented). <field value> is a list of message ids.
8:	Subject. (**Subject:** field). <field value> is a text string.
9:	ExpiryDate. (Not implemented). <field value> is a UTC date string.

Table A-2 Continued

Field Number	Description and Field Value
10:	ReplyBy. (Not implemented). <field value> is a UTC date string.
11:	ReplyToUsers. (Not implemented). <field value> is a list of mailbox names.
12:	Importance. (Not implemented). <field value> is one of the following: 0 Low 1 Normal 2 High
13:	Sensitivity. (Not implemented). <field value> is one of the following: 1 Personal 2 Private 3 Company Confidential
14:	AutoForwarded. (Not implemented).

A.3 Profile File Format

The Profile file is in the SYSTEM:/MAIL directory. Its filename is the same as the name of the user who ran the PROFILE command to create it. It has the extension PRF.

The Profile file consists of a number of lines of text, each of which contains a keyword followed by a colon, white space and the value of the specified variable.

Not all keywords need be included in the file, and THE FILE NEED NOT EXIST. If the file exists and a keyword is omitted, it will have the default value. Blank lines are allowed and ignored.

Table A-3 shows each profile keyword and its description.

Table A-3. Keyword and Description in Profile File

Keyword	Description
PROMPT	Value is the string to use for mail system prompt. The default prompt is MAIL>.
UNAME	Value is the user's personal name as he wants it to appear in the From: field of messages.
DOMORE	YES or NO indicates if, when READing a message, the display should pause after a screenful of text and wait for the user to type a key.
INCSELF	YES or NO indicates whether the user wishes to receive copies of his reply messages.
ASKCC	YES or NO indicates if the user wants to be asked for a list of mailboxes that are to receive courtesy copies of the message he is SENDing.
ASKSUBJ	YES or NO indicates if the user wants to be asked to supply a subject field for the message he is SENDing.
OUTPFF	YES or NO indicates if the user wants messages OUTPUT to a file to be separated by a form feed (YES) or just a line feed (NO).
SHOWNEW	YES or NO shows whether the user wants the mail system to list NEW messages automatically when he starts up.
ASKFLUSH	YES or NO indicates whether the user wants to be asked if deleted messages should be removed from his mailbox when he issues an EXIT command (YES), or if they should be removed without asking (NO).

A.4 Pipe Message Format

Messages sent through the data pipe to the Mail Delivery Process must be in the following format:

- The message must consist of a number of variable-length ASCII text lines terminated with carriage return/line feed.
- The message must end with a line containing a single period.
- Any line in the message text that begins with a period must have an extra period prefixed to it. The Mail Delivery Process removes the leading period from any line that begins with two or more periods.

The following example shows a message in the pipe message format:

```
Sender's mailbox name<CRLF>
Expiration date of message (see below)<CRLF>
Number of recipients<CRLF>
First recipient's mailbox name<CRLF>
Second recipient's mailbox name<CRLF>
      .
      .
      .
Last recipient's mailbox name<CRLF>
Text of mail message
. <CRLF>
```

A.4.1 Expiration Date

The Mail Delivery Process tries to deliver a queued message for a period of one week from when the message was sent, after which time a message is considered undeliverable.

The expiration date is in the following format:

YYMMDDHHMMSS+-HHMM

All fields have a leading zero if the value is less than 10.

The date displays local time, plus or minus the offset from the Universal Time Coordinate (UTC).

From left to right, the letters indicate the following:

- last 2 digits of the year
- two digits for the month
- two digits for the day
- two digits for hours, local time
- two digits for minutes, local time
- two digits for seconds, local time
- hours offset from Universal Time Coordinate
- seconds offset from Universal Time Coordinate

A.5 Queued Message File Format

The Mail Delivery Process copies messages from the data pipe into Queued Message Files. These files have the extension QML and are placed in the directory SYSTEM:MAIL/OUTGOING.

The queued message format is similar to the pipe message format. Note that the file starts with a 12-byte fixed width decimal field that indicates the start of the message text.

The following example shows a message file in the queued message format:

```
Offset of start of text <CRLF>
Sender's mailbox name<CRLF>
Expiration date <CRLF>
Number of recipients<CRLF>
<STATUS>First recipient's mailbox name<CRLF>
        .
        .
        .
<STATUS>Last recipient's mailbox name<CRLF>
Text of message (to EOF)
```

Valid values for <STATUS> are:

" " (Space)	Message has not yet been delivered to this user.
"-"	Message has been delivered to this user or forwarded to the destination node.

Number of Recipients is a 4-byte fixed-width field. The number can be different from the number in the pipe message because the Mail Forwarding Data Base file is read and can contain an alias name with many users in a list.

A.6 MAIL Message Fields

MAIL supports message fields described in the International Telegraph and Telephone Consultative Committee (CCITT) 1985 Red Book, Fascicle VIII.7, "Data Communication Networks Message Handling Systems," Recommendations X.400 - X.430.

MAIL supports the following required fields. The field name that MAIL displays to the user appears in parentheses.

- Message ID (Message-id:)
- Send Date (Date:)
- Originator (From:)
- Primary Recipient(s) (To:)

MAIL supports the following optional fields:

- Copy recipients (Cc:)
- In Reply to (In-reply-to:)
- Subject (Subject:)

MAIL does not implement any other field, but the field will be displayed if it is received in a message.

End of Appendix A

1082-2008-001

FlexNet™
Network Operating System
OEM and Programmer Guide

1082-2023-001

First Edition: November, 1986

Software Version: 1.3

Foreword

FlexNet™ is a network software package for FlexOS™ that provides distributed access to remote computer resources. The resources available for shared access are disks, printers, and pipes. In addition, FlexNet provides a programming interface to the network itself.

You access remote resources using the standard set of FlexOS Supervisor calls. FlexNet intercepts calls for remote resources and transfers them to a proxy process on the server. (FlexNet creates and manages the proxy processes on each server.) This process makes the call to the Supervisor on behalf of the requesting process and returns the data and return code. The entire operation, from trapping the call to the return of the return code is transparent to the calling process.

FlexNet, like FlexOS, is portable among computers with different microprocessors and allows machines with different processors to communicate on the same network. FlexNet also includes drivers that provide file system access from PC DOS requesters and to PC DOS servers.

FlexNet software consists of a FlexOS resource manager and a set of loadable drivers. The Network Resource Manager installs the drivers and performs requester functions. The network drivers provide name service, server functions, and the device-style interface (like a physical device) that you call to access the network directly. The degree of network services provided is consistent with the Application through Session layer functions specified by the International Standards Organization (ISO) model for Open System Interconnection.

To complete the communication package, FlexNet requires an OEM to provide the network hardware and software driver to perform ISO transport through physical layer services. Because the ISO levels 1 through 3 are functionally independent of its upper layers, the network hardware can take any form. Note, however, that FlexNet requires the network drivers to provide virtual circuit services and guaranteed message delivery consistent with the ISO level 4, 8073 standard.

Intended Audience

This book is written for two audiences: the original equipment manufacturer creating a network interface and the programmer responsible for writing network applications or utilities. The OEM should be well-versed in network technology and familiar with the architecture and driver conventions of FlexOS. The programmer should be familiar with FlexOS Supervisor calls (SVCs) and calling conventions. The descriptions of both the network SVCs and the driver interfaces assume the reader has a knowledge of C programming techniques.

FlexNet Documentation

The FlexNet documentation set consists of this manual and the FlexNet Network Operating System User's Guide (referred to as the FlexNet User's Guide).

The FlexNet User's Guide has two distinct parts. The first part describes the user interface; the second describes on-site network and name service installation for the network system manager.

The information presented in this manual builds upon that presented in the FlexOS documentation set. The FlexOS documentation set consists of the following manuals:

- FlexOS System Guide: This book describes the FlexOS system design, driver model, and driver services. Develop your FlexNet drivers according to the information presented in this book.
- FlexOS Programmer's Guide: Refer to this book for its description of the FlexOS calling conventions, Supervisor calls, and system tables.
- FlexOS User's Guide: Read this book for its description of the FlexOS utilities and system manager functions.

Section Contents

The information in this book is organized into the follow sections:

- Section 1 presents an overview that describes FlexNet components, features, and operating conventions.
- Section 2 describes the FlexNet Supervisor calls.
- Section 3 describes the FlexNet tables managed by the Network Resource Manager.
- Section 4 describes the transport driver and name service driver interface.
- Section 5 describes network configuration options.
- The appendixes describe the FlexNet message format and contents, PC DOS requester and server support, and the network error codes.

Contents

Tables

Figures

Listings

FlexNet Overview

This section describes the FlexNet components and the degree of FlexOS file system support provided by the network. This information is relevant to both the FlexNet application programmer and the OEM.

1.1 FlexNet Components

FlexNet provides a network interface for the FlexOS operating system. Figure 1-1 illustrates the FlexNet components and their relationships with the FlexOS Supervisor and the physical network.

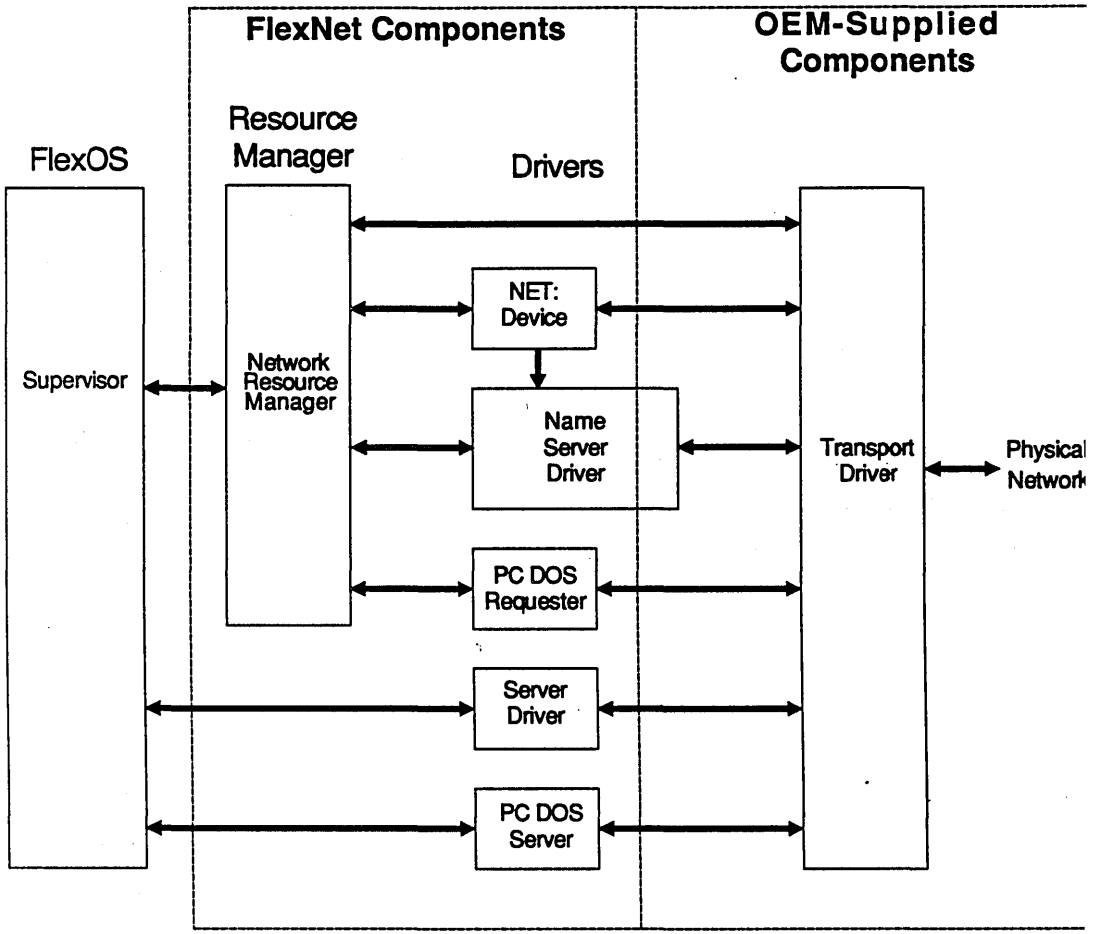


Figure 1-1. Network Interface Components

The components enclosed by the dashed lines in Figure 1-1 are provided with FlexNet. The Transport Driver and hardware to the right of the enclosure are not provided. Notice that the Name Server Driver straddles the dashed line. The FlexNet Name Server Driver provides

basic node name-to-address translation. However, it requires you to define the node name and address conventions.

Table 1-1 describes the FlexNet components. All drivers are dynamically loadable. A requester node requires only the Network Resource Manager, the Name Server Driver, and the Transport Driver. Server nodes require the Network Resource Manager (thus making them inherently requesters as well), the Name Server Driver, the Server Driver, and the Transport Driver. The NET: device is necessary only if you write programs that require direct access to the network.

Table 1-1. FlexNet Components

Component	Description
Network Resource Manager	FlexNet's resource manager manages the requester interface to the Transport Driver, the NET: device, the Name Server Driver, and the PC DOS Requester. It performs name and address translation for the Transport Driver and PC DOS Requester but not for the NET: device.
NET: Device	The NET: device driver provides direct access to the network. Use it to make internode connections, send and receive messages through a connection, and send and receive datagram (connectionless) messages. It handles name and address translation through the Name Server Driver independently of the Network Resource Manager.
Name Server Driver	FlexNet's name service translates logical node and socket names to their corresponding network address for the Network Resource Manager and NET: device. The name database is disk-based on each node. A subset of the database entries can be loaded into memory. See Section 4.2 for the description of the Name Server Driver and how you modify it to accommodate your network's addressing scheme.

Table 1-1. (Continued)

Component	Description
PC DOS Requester	The PC DOS Requester Driver formats messages into the SMB PC LAN dialect for output to a system running the IBM®PC Network Program. See Appendix B for the description of the FlexOS functions supported by this driver.
Server Driver	The Server Driver manages a set of processes which make Supervisor calls (SVCs) on behalf of a remote process. This proxy process assumes the identity of the remote process for servicing call. Once the call is completed, the process is available to represent another remote process.
PC DOS Server	The PC DOS Server Driver interprets SMB messages formatted according to the PC LAN Program dialect. See Appendix B for the description of the SMB functions supported.
Transport Driver	The Transport Driver is the OEM-developed driver that performs the network's transport through physical layer functions. FlexNet requires the Transport Driver to provide ISO level 4 virtual circuit services and interfaces readily to existing 8073-type products. See Section 4.1 for the description of the transport driver interface.

The Network Resource Manager and Server Driver work together as requester and server to provide the remote processing of file system and table-related SVCs. You identify the remote device to be accessed by placing its node name at the beginning of the path specification. For example, the disk file GL.DAT on network node MARY drive c: would be referenced by

MARY::c:/GL.DAT

FlexOS node names are always delimited in path specifications by double colons.

Access to remote devices and files (including pipes) is performed according to standard FlexOS procedures. For example, to use the GL.DAT file, you make an OPEN call and use the file number returned for all subsequent operations. Not all SVCs are supported over the network, however. Table 1-2 lists the calls supported.

The Network Resource Manager and the Server Driver together provide a transparent network interface over which you do not have direct control. Alternatively, you make direct use of the network through the NET: device. Use this interface to establish connections with remote nodes, send and receive messages, and send and receive datagrams. Use NET: as you do any device--you open it first and use the file number returned for subsequent operations. The program interface to NET: consists of the standard FlexOS OPEN, CLOSE, and CANCEL SVCs and a set of SPECIAL SVCs. The NET: interface is described in Section 2.

FlexNet does not limit the number of nodes per network nor require a particular node addressing scheme or network topology. All nodes are inherently network requesters; the network server function is added by installing the Server Driver.

FlexNet formats messages to FlexNet servers according to the IBM Server Message Block (SMB) protocol using a dialect called **FLXNET1.3**. The SMB format is described in Appendix A .

In communication with IBM PC LAN Program Servers, the FlexNet PC DOS Requester Driver formats the messages according to the SMB PC LAN Program 1.0] dialect. The FlexNet PC DOS Server Driver interprets messages in this format as well. The NET: Device Driver does not format the message according to any protocol.

DR NET adds the following operating system utilities:

NAMES	Displays and modifies entries in the Name Server Driver's memory-based translation table. (The name service database file is created and modified with an editor.)
NETSET	Selects network installation options
NETSTAT	Displays current connection information

FlexNet also makes use of FlexOS's LOGON and LOGOFF command. See the FlexNet User's Guide for the description of these utilities. NETSET options are also described in Section 5.

1.2 Access Conventions

Network communication is either connection-oriented or connectionless. Connection-oriented communication provides guaranteed message delivery. Connectionless communication, referred to as datagrams, does not guarantee message delivery.

1.2.1 Connections

A connection is a virtual circuit between specific sockets on the same or different nodes. Nodes and sockets are identified by network addresses. Make the name-to-address assignment for all sockets you require in the FlexNet Name Server Driver's NETNAMES.DAT resource file.

All communication between the Network Resource Manager and the Server Driver is connection-oriented. The sockets used are named DRIRQSTR and DRISRVR. These sockets are reserved for FlexNet Network Resource Manager and Server Driver use only and must have the same address on each node. Section 4.2 describes the reserved socket and node names and how to define them.

The NET: Device Driver offers both connection-oriented and datagram communication. Using a set of SPECIAL SVCs, you make and break connections and send messages.

The total number of simultaneous connections allowed by FlexNet is a

settable network parameter. The maximum applies to the total connections established through the Network Resource Manager, Server Driver, and NET: Device Driver. Section 5 describes how to set this and other network parameters.

Figure 1-2 illustrates network connections.

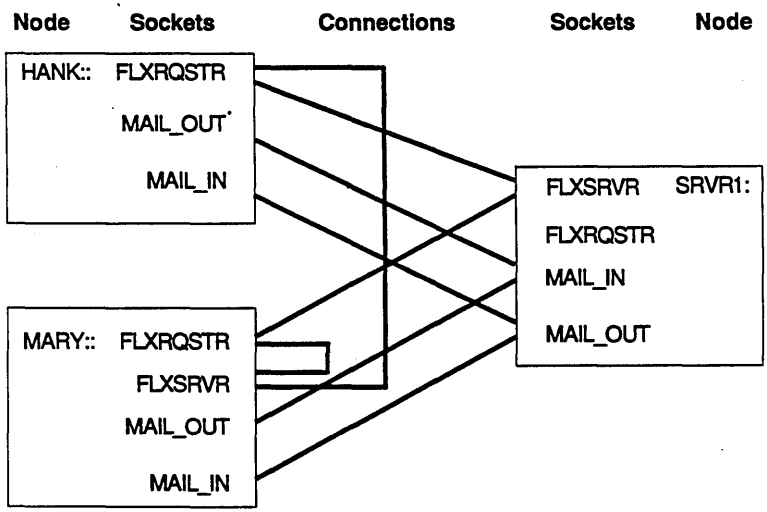


Figure 1-2. Node and Socket Connections

The sample configuration in Figure 1-2 illustrates some FlexNet characteristics and some OEM-dependent features. DRIRQSTR and DRISRVR are the sockets used by the Network Resource Manager and Server Driver. The DRISRVR is only present on nodes with the Server Driver installed. Notice that sockets can have multiple connections through it. Also notice that you can have a connection between different sockets on the same node.

The MAIL_IN and MAIL_OUT sockets illustrated are not provided with FlexNet. They are shown to demonstrate a use of the NET: device: for an electronic mail package. In the figure, nodes HANK:: and MARY:: have connections between their MAIL_IN and MAIL_OUT sockets and the SRVR1::'s MAIL_OUT and MAIL_IN sockets. This allows connection-

based communication between these nodes but not between HANK:: and MARY::. This does not preclude communication between HANK:: and MARY::, however. The application can use the NET: device's datagram services to send and receive messages. Note that a node does not have to have the Server Driver installed to receive messages, you can wait on a socket for a message using the NET: device.

Users make and break connections to servers by logging on and off. On servers where FlexOS protection is enabled, a settable network parameter called the security mode determines whether an explicit log on is required or an implicit log on is provided. See Section 1.2.3 for the description of security modes. For both explicit and implicit log ons, the user must have a USER.TAB account on the server to gain access.

On servers where FlexOS protection is not enabled, implicit log on is provided by FlexNet and all users have access to all resources on the node.

Node Names

Node names can be up to 15 bytes long, must begin with a letter, and must contain only letters or numbers. Depending on the Name Server Driver implementation, characters are either forced to uppercase or case-sensitive. In the remainder of this manual, we assume the names are case-sensitive.

The node name must precede all other device and file references in the path specification and is always terminated by two colons. For example, the following path identifies the file COA.DAT in the subdirectory GL on drive c: on node ACCOUNTING.

```
ACCOUNTING::c:\GL\COA.DAT
```

Node names are set in the name service database file. You can create the name database file with a text editor. The text editor must not put special characters in the file.

The local node name is established with the NAMES utility and should be done in the computer's CONFIG.BAT file. The computer cannot function as a requester or server until its node name has been set.

Sockets

Sockets identify a node's logical connection points. Socket names have a 8-character limit, must start with a letter, and are restricted to letters and numbers. Depending on the Name Server Driver implementation, characters are either forced to uppercase or case-sensitive. In the remainder of this manual, we assume the names are case-sensitive.

FlexNet requires two sockets for requester-to-server communication: DRIRQSTR and DRISRVR. The Network Resource Manager sends messages through DRIRQSTR and listens on it for. the response; the Server Driver listens on DRISRVR and sends responses through it. Do not use these socket for applications.

Each node can have multiple sockets, and you can establish multiple connections through a single socket. FlexNet imposes no limit to the number of sockets you can have at one time.

1.2.2 Server Processes

The FlexNet Server Driver creates at least two server processes when it is installed; one to handle all synchronous SVCs and the other to handle all asynchronous calls from remote nodes.

The Server Driver assigns a server process to a remote process when a call is received. The server process assumes the identity of the remote process by putting the remote process's node, process, and family IDs (these are acquired from the message header) in its ENVIRON table's RNID, RPID, and RFID fields. The message's parameter and buffer data are then used to form the SVC and its parameter block. The server process waits for the SVC to complete and sends a response message with the return code and data.

The relationship between remote and proxy processes expires once the call has been made. At this point, the server process becomes a free agent, available to service a call from any requester. If no server process is available when a message is received, the message is registered in a FIFO queue for servicing when a process becomes free.

Note: The Server Driver is not open for modification. However, you can adjust the number of synchronous server processes and the number of simultaneous asynchronous events. See Section 5 for the instructions for setting these network parameters.

1.2.3 Remote Accounts and Security Modes

User access to FlexOS servers is controlled using a combination of FlexOS's user accounts and access privileges and FlexNet's server security mode. Before a user can log on a remote computer, he or she must have either an account on the server in his or her name or an account for which he or she knows the name and password.

The user's access procedure to the node is governed by the server security mode selected. There are two modes:

Normal — Log on is automatic upon first access when the local and remote account names match.

Extended — Explicit log on with the LOGON utility is always required.

Extended and normal security modes apply only if FlexOS protection is set on the server node. If protection is off, all users have ungoverned access to all devices on the server.

When FlexOS protection is set on requester and server nodes, access is granted automatically under normal security when the user making the request has an account (that is, USER.TAB entry) with the same name on the server node. Note that the accounts on the two nodes can have different user and group ID numbers. The user and group IDs valid on each node are the values from its machine's USER.TAB file.

To use an account with a different name, the user must invoke the LOGON utility and specify the node name in the command line. LOGON prompts the user to enter the account name and password. Users can log on only one remote account at a time. If the user is logged on a remote account and then successfully logs on another, FlexNet automatically logs off the original account.

Under extended security, the Server Driver requires explicit log on at all times. The user must enter the LOGON command and supply the remote user name and password when prompted before access is allowed.

The NETSET utility allows you tp select the security mode for each account. You can have different modes on separate servers but you cannot mix modes on a single node; normal or extended security

applies to all accounts. See Section 5 for the instructions for setting the security mode.

Note: Access to PC DOS servers is governed by different rules. See the LOGON table description in Section 3 and Appendix B for details.

1.2.4 Watchdog Process

The Network Resource Manager creates a watchdog process that issues periodic requests, referred to here as echo messages, to all servers with which there is a connection. Similarly, the Server Driver has a process that waits for a echo message from each connection and issues a response when one arrives.

The requester and server watchdog processes assume the connection is broken when a response is not received or the echo message does not arrive within a specific time period. The responses to a broken connection are as follows:

- On the requester, the Network Resource Manager cancels all operations in progress with that node.
- On the server, the Server Driver cancels all outstanding requests from and closes all files opened by processes using that connection.

You can set two watchdog parameters, the regularity of the echo message and the time period within which the server must receive the echo. Section 5 describes how to set these parameters.

1.3 FlexOS SVC Support

FlexNet supports the remote processing of most FlexOS file and table management SVCs, providing access to remote disk, pipe, and device files. Remote console files are not accessible.

Table 1-2 lists the FlexOS SVCs and indicates whether the call can be processed remotely and, if so, whether or not the asynchronous mode is supported.

Note: Observe the following precautions in your READ, WRITE, and LOCK calls.

READ	FlexNet breaks down large files into blocks for transfer over the wire. This allows another process to overwrite file data before the entire transfer completes. Call LOCK to lock records from access by other processes during the transfer and then to unlock the records when the transfer is complete.
WRITE	As above, call LOCK to prevent overwriting.
LOCK	You cannot cancel an asynchronous attempt to unlock a file or portion of a file that is on a remote system.

For asynchronous READ and WRITE calls, do not reuse the buffer until the event has completed.

Table 1-2. FlexNet Supervisor Call Support

SVC	Processed Remotely?	Async	SVC	Processed Remotely?	Async
ABORT	NO		LOCK[3]	YES	YES
ALTER	NO		LOOKUP	YES	NO
CANCEL	YES	NO	MALLOC	NO	
CLOSE	YES	NO	MFREE	NO	
COMMAND[1]	NO		OPEN	YES	NO
CONTROL	NO		ORDER	NO	
COPY	NO		READ[3]	YES	YES
CREATE[2]	YES	NO	RENAME	YES	NO
DEFINE	NO		RETURN	NO	
DELETE	YES	NO	SET	YES	NO
DISABLE	NO		SPECIAL[4]	YES	YES
DEVLOCK	YES	NO	STATUS	NO	
ENABLE	NO		SWIRET	NO	
EXIT	NO		TIMER	NO	
GET	YES	NO	WAIT	NO	
GIVE	NO		WRITE[3]	YES	YES
INSTALL	NO		XLAT	NO	
KCTRL	NO				

Notes:

1. The file specified in COMMAND can be loaded from a remote disk. However, you cannot create a process on a remote node.
2. CREATE is valid only for remote disk files, directories, and pipes; you cannot create a remote virtual console.
3. See the note above regarding the network use of these calls.
4. SPECIAL calls that might be processed remotely must not contain address pointers in the data or parameter buffers.

End of Section 1

Program Interface

The program interface to FlexNet consists of a group of Supervisor calls (SVCs) that control the NET: device and a set of network tables. Direct access to the network is provided through the NET: device using the standard OPEN and CLOSE SVCs and several SPECIAL SVCs. Use the SPECIAL calls to establish connections between NET: devices on nodes and to send or receive connection-oriented or connectionless messages.

This section describes NET: calling conventions and the SPECIAL calls used to access it.

2.1 NET: Access Conventions

To use the network explicitly, make an OPEN call to open the NET: device. As with other devices, a file number is returned which you use for all subsequent calls to the NET: device.

Use the standard CLOSE call to disassociate the file number from the NET: device and the standard CANCEL call to terminate an asynchronous event. The remainder of the NET: interface consists of a set of SPECIAL calls.

Internode communication is either connection-oriented or connectionless. To have guaranteed end-to-end message delivery, you must first establish a connection between specific sockets on each node. The connection is assigned a unique number, referred to as the handle. Use the handle to specify the connection in all subsequent communication calls. You should disconnect the connection when you no longer require it.

Connectionless communication, hereinafter referred to as datagrams, does not require an established connection. Datagram communication further differs from connection-oriented communication in that message reception is not guaranteed and the message size is limited to the network hardware's message buffer size.

FlexNet provides node and socket name translation for connection-oriented and datagram communication. However, it does not package the message according to the SMB format. For all communication through the NET: device, you define the message format.

2.1.1 Network Address Buffer

The Network Address Buffer (NAB) is the data structure you use to designate source and destination node and socket names in your calls to the NET: device. The data structure consist of four separate buffer specifications for the logical names. Figure 2-1 illustrates the NAB format. Although you are not always required to specify all node and socket names, you must always provide adequate buffers for each. FlexNet uses these buffers to fill in the names omitted before the call returns. All names must be unambiguous--wildcards are not allowed.

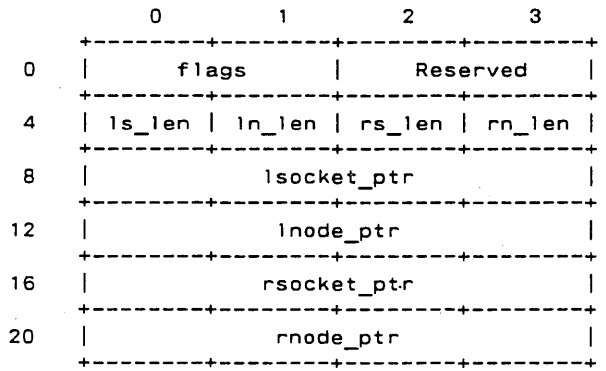


Figure 2-1. Network Address Buffer

The NAB fields are defined as follows:

flags — Bit map of flags assigned as follows:

Bit 0: 0 = Local socket name is provided
1 = Local socket name is not provided

Flag bits 1 through 15 must be set to zero.

ls_len	Size in bytes of the local socket name buffer
ln_len	Size in bytes of the local node name buffer
rs_len	Size in bytes of the remote socket name buffer
rn_len	Size in bytes of the remote node name buffer
lsocket_ptr	Pointer to the local socket name buffer
lnode_ptr	Pointer to the local node name buffer
rsocket_ptr	Pointer to the remote socket name buffer
rnode_ptr	Pointer to the remote node name buffer

The NET: device automatically calls the Name Server Driver to translate the node and socket names to physical network addresses before it calls the Transport Driver. The NET: Device and Network Resource Manager use the same database file.

2.1.2 Connection Handle

The connection handle is returned in the low-order word from SPECIAL calls to NET: that request connections and wait for connection requests from other nodes. Use the handle in NET: send and receive calls to identify a specific connection. Do not use the connection handle to send and receive datagrams.

Connection handles are valid for the file number specified in the SPECIAL call that generated it only. This does not, however, preclude opening the NET: device multiple times. If you do open it several times, be sure to keep the affiliation between file numbers and handles straight. Attempts to use a handle returned under a different file number results in an error.

2.1.3 Timeout Parameter

All NET: calls except DG_SEND provide a timeout parameter so that you can limit the amount of time NET: waits for the function to complete. The timeout value is a 16-bit unsigned integer indicating 100 millisecond time increments. If a value larger than 0xFFFFH is specified, 0xFFFFH is assumed. To specify no time limit, set the timeout value to zero.

2.2 NET: Device SVCs

This section describes the SPECIAL calls used to access the NET: device. Refer to the FlexOS Programmer's Guide for the description of the OPEN, CLOSE, and CANCEL calls and the generic description of the SPECIAL parameter block input parameters. Note that the OPEN and CLOSE flags are not used by the NET: device and must be zero. Table 2-1 lists the NET: SPECIAL calls. The call descriptions are presented in alphabetical order.

Table 2-1. NET: SPECIAL Supervisor Calls

Name	Number	Function
AWAIT_CONNECT	C2H	Wait for a connection request
CONNECT	C1H	Send a connection request
DISCONNECT	C5H	Terminate a connection
RECEIVE	84H	Wait for a message on a specific connection
SEND	C3H	Send a message over a specific connection
DG_SEND	C6H	Send a datagram to a node and socket
DG_RECEIVE	87H	Wait for a datagram on a node and socket

The C interface and the parameter block for all SPECIAL calls is as follows:

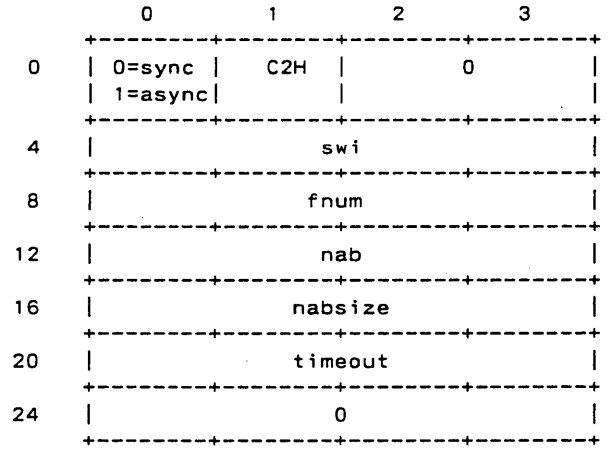
```
UWORD     flags;
LONG      fnum,dbufsize,pbufsiz,emask;
BYTE      func,*databuf,*parmbuf;

ret = s_special(func,flags,fnum,databuf,dbufsiz,parmbuf,pbufsiz);
emask = e_special(swi,func,flags,fnum,databuf,dbufsiz,parmbuf,pbufsiz);

ret = __osif(F_SPECIAL,&parmblk);
```

The SPECIAL parmblk format for each NET: SPECIAL call is shown with the call's description. Appendix C contains the network-specific return codes.

None of the NET: SPECIAL calls make use of the flags parameter. For all calls, this value should be 0. NET: supports synchronous and asynchronous access and the use of software interrupts; use the swi parameter as the pointer to your interrupt routine. In all of the SPECIAL calls, enter the file number returned from your OPEN NET: call as the fnum value.

The Return Code in each description indicates the value returned by the synchronous form of the function. If you use the asynchronous form, you get an event mask instead. Use the RETURN SVC to get the return code after the event has completed.

2.2.1 AWAIT_CONNECT



Parameters

nab	Pointer to the Network Address Buffer. Provide local node and socket names and buffers for the remote node and socket names.
nabsize	Size in bytes of the Network Address Buffer.
timeout	Timeout period to wait for connection

Return Code

handle	Connection number for the virtual circuit
error code	See Appendix C for the network error codes

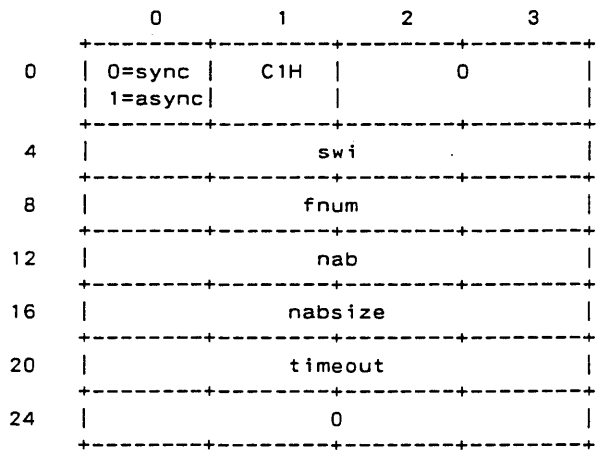
Description

Use AWAIT_CONNECT to wait for a connection request. Put the local node name in the NAB lnode buffer and the socket name to listen on in the lsocket buffer. Provide buffers for the remote node and socket names with the rsocket_ptr and rnode_ptr pointers. Set rs_len and rn_len to the maximum buffer size.

The return code is returned when a connection request for the socket specified is received or the timeout occurs. The return code contains the 16-bit handle in its low order word.

On return the connection's remote node and socket names are recorded at the buffers you indicated. The rs_len and rn_len values indicate the actual socket and node name lengths.

2.2.2 CONNECT



Parameters

nab	Pointer to the Network Address Buffer. Provide the local node and remote node and remote socket names. The local socket name is optional. However, you must provide a socket name buffer. Set flag bit 0 to 0 if you provide the socket name or to 1 if you want DR Net to pick a socket.
nabsize	Size in bytes of the Network Address Buffer
timeout	Timeout period to establish for connection

Return Code

handle	Connection number for the virtual circuit
error code	See Appendix C for the network error codes

Description

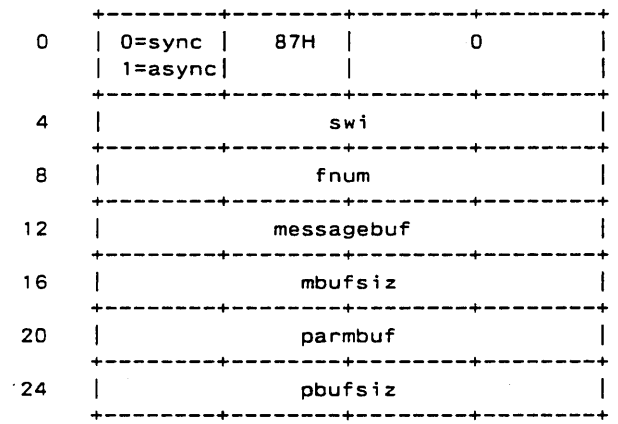
Use CONNECT to establish a connection to the remote node and socket specified in the Network Address Buffer. FlexNet does not require you to specify the local socket. If you do not, set flag bit 0 and provide a buffer at the lsocket_ptr. Set the maximum local socket name length in ls_len.

The call returns with the connection handle in the low order word of the return code or an error code. When you do not specify a local socket, FlexNet picks one for you, records the socket name lsocket_ptr address, and sets the ls_len value to the exact name length.

The remote node must have an outstanding SPECIAL AWAIT_CONNECT call on the designated socket for the CONNECT to succeed. If the AWAIT_CONNECT has not been issued, FlexNet keeps trying to establish the connection until the timeout period expires.

You establish network connections to your own node by specifying the local node name and a local socket as the remote destination.

2.2.3 DG_RECEIVE



Parameters

messagebuf	Pointer to data buffer in which to receive message
mbufsiz	Size in bytes of the message buffer
parmbuf	Pointer to parameter buffer. The parameter buffer contains the Network Address Buffer (NAB) followed by a 16-bit unsigned timeout value. The NAB must provide local node and socket names and allocate buffers for the remote node and socket names.
pbufsiz	Size in bytes of the parameter buffer

Return Code

no_bytes	Number of bytes received
error code	See Appendix C for the network error codes

Description

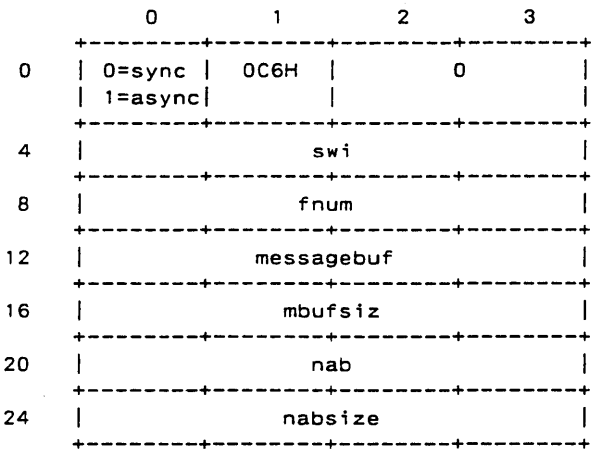
Use DG_RECEIVE to receive a datagram. In your call, you provide pointers to the message receive buffer and to a parameter buffer. The parameter buffer must contain a Network Address Buffer (NAB) structure followed by a 16-bit unsigned timeout value for the operation.

The maximum buffer size for datagrams is the size of the network hardware's message buffer. FlexNet cannot send multipacket datagrams. The size of the hardware message buffer differs from one network implementation to another.

Required parameters in the Network Address Buffer are the local node and socket names and remote node and socket buffers. Set the rs_len and rn_len to the maximum buffer size.

This call returns with the number of bytes received in the return code. In addition, FlexNet fills the rsocket and rnode buffers and sets the rs_len and rn_len values to the actual name lengths.

2.2.4 DG_SEND



Parameters

messagebuf	Pointer to buffer with message to be sent
mbufsize	Size in bytes of the message buffer
nab	Pointer to Network Address Buffer (NAB). Provide all local and remote node and socket names.
nabsize	Size in bytes of the Network Address Buffer

Return Code

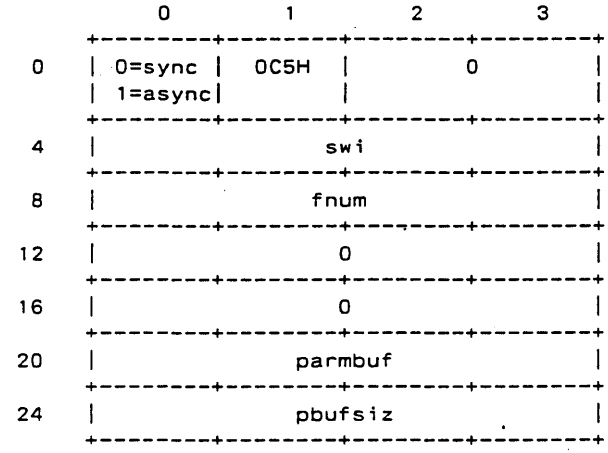
E_SUCCESS	Operation was a success
Error code	See Appendix C for the network error codes

Description

Use DG_SEND to send a datagram. Put the message in the buffer at the messagebuf pointer and the local and remote node and socket names in the Network Address Buffer. There is no timeout parameter for DG_SEND.

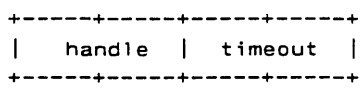
The maximum buffer size for datagrams is the size of the network hardware's message buffer. FlexNet cannot send multipacket datagrams. The size of the hardware message buffer differs from one network implementation to another.

2.2.5 DISCONNECT



Parameters

parmbuf — Pointer to a 4-byte data structure formatted as follows:



where handle is the connection number to terminate and timeout is a 16-bit unsigned value.

pbufsiz — Size in bytes of the parameter buffer

Return Code

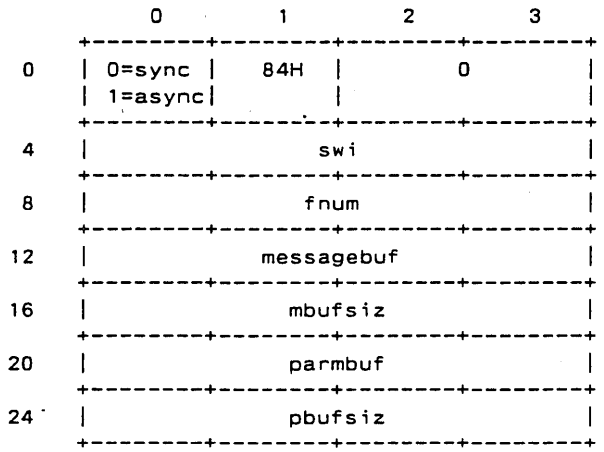
E_SUCCESS — Operation was a success

Error code — See Appendix C for the network error codes

Description

Use DISCONNECT to terminate a connection established through the NET: Device. Indicate the connection in the parameter buffer by its handle and set a time limit for the operation to complete. FlexNet waits for all SEND calls to complete before terminating the connection. Outstanding RECEIVE calls are aborted.

2.2.6 RECEIVE



Parameters

messagebuf	Pointer to the receive message buffer
mbufsize	Size in bytes of the message buffer
parmbuf	Pointer to a 4-byte data structure indicating the connection handle and time limit for the operation. See the DISCONNECT description for the parameter buffer format.
pbufsiz	Size in bytes of the parameter buffer

Return Code

no_bytes	Number of bytes received in last buffer
Error code	See Appendix C for the network error codes

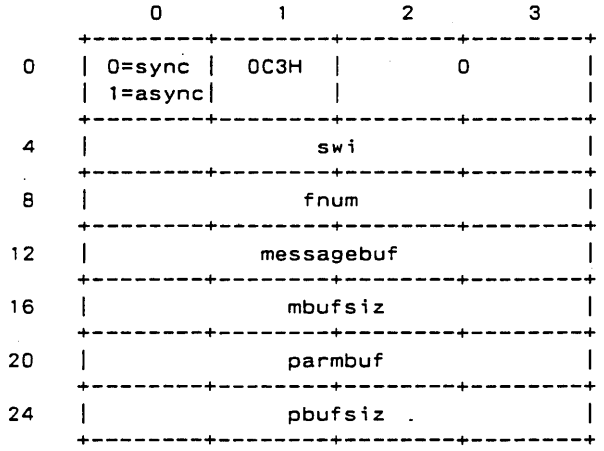
Description

Use RECEIVE to wait for a message on an established connection. Indicate which connection in the parameter buffer by its handle. Also put the timeout value in the parameter buffer. Set the timeout to zero for no limit.

When the message size exceeds the buffer size, you receive the E_MOREDATA return code. To get the next block, make another RECEIVE call. If the remainder fits in the buffer, you get the success return code. If there is still more message, you get another E_MOREDATA error. Repeat RECEIVE calls until the success return code is provided.

The success return code indicates the number of data bytes in the last message buffer.

2.2.7 SEND



Parameters

messagebuf	Pointer to the buffer with the message data
mbufsize	Size in bytes of data buffer
parmbuf	Pointer to a 4-byte data structure indicating the connection handle and time limit for the operation. See the DISCONNECT description for the parameter buffer format.
pbufsiz	Size in bytes of the parameter buffer

Return Codes

E_SUCCESS	Operation was a success
Error code	See Appendix C for the network error codes

Description

Use SEND to send a message over an established connection. Indicate which connection in the parameter buffer by its handle. Also set the time limit in the parameter buffer. SEND guarantees message delivery unless an error is returned.

The message length is independent of the receiving node's message buffer size. FlexNet breaks down the message into manageable blocks at the destination node. See the RECEIVE description for the explanation of this procedure.

End of Section 2

Network Tables

The Network Resource Manager supports the data structures listed in Table 3-1. As in other FlexOS tables, the network tables are manufactured in response to certain SVCs and do not actually exist as reserved blocks in memory.

Table 3-1. Network Resource Manager Tables

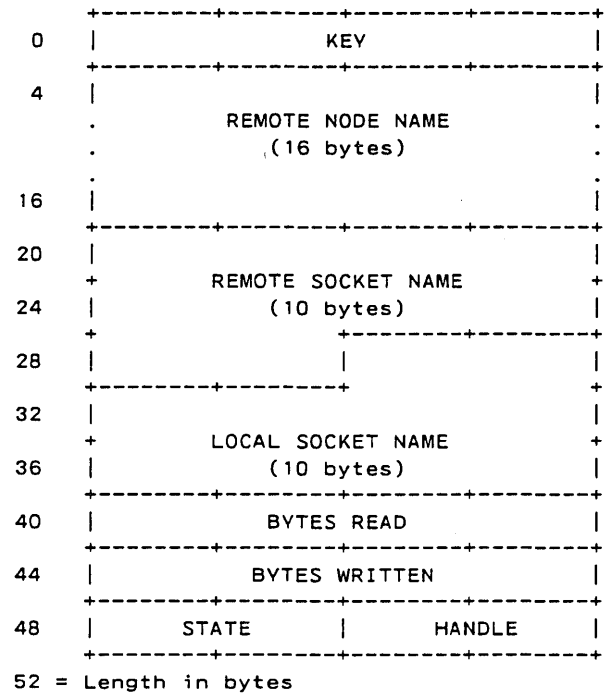
Table Number and Name	GET	SET	ID	LOOK UP	Key	Purpose
60H NETPARAM	X	X				Get and set network parameters
61H CONNECTION				X	key	Show status of connections
62H LOGON		X	0	X	key	Log on and show remote account information
63H NETNAME		X	0	X	key	Get and set in-memory names database

The CONNECTION table is available only through the LOOKUP SVC while the LOGON and NETNAME tables are available through the LOOKUP and SET calls. The Network Resource Manager keeps only one copy of the NETPARAM table so it does not require an ID specification in your GET and SET calls.

CONNECTION, LOGON, and NETNAME table descriptions follow below. See Section 5 for the description of the NETPARAM table.

3.1 CONNECTION Table

The CONNECTION table indicates the status of a virtual circuit. CONNECTION tables are available only through the LOOKUP SVC--you cannot use GET to retrieve a single table nor SET to change a table parameter. Use the remote node name as LOOKUP's name parameter. Names are always uppercase. Use the * wildcard to retrieve multiple tables.



- **KEY:** Key field for LOOKUP
- **REMOTE NODE NAME:** Null-terminated name of the connection's remote node. This field is null when the local socket is in the LISTEN state.
- **REMOTE SOCKET NAME:** Null-terminated name of the connection's remote socket. This field is null when the local socket is in the LISTEN state.

- **LOCAL SOCKET NAME:** Null-terminated name of the local socket
- **BYTES READ:** Number of bytes received through the connection since it was established
- **BYTES WRITTEN:** Number of bytes sent through the connection since it was established
- **STATE:** A value indicating the current connection state of the socket. Table 3-2 lists the connection state values.
- **HANDLE:** The connection number returned by CONNECT and AWAIT_CONNECT calls.

Table 3-2. Connection States

Value	State	Description
0	IDLE:	The socket has no connections established.
1	CONNECT:	A connect request has been issued to the Transport Driver but the virtual circuit has not yet been established.
2	VERIFY:*	A virtual circuit is established, however, dialect resolution is in progress.
3	Reserved	
4	OPEN:	The connection is established over the virtual circuit and available for use.
5	CLOSING:	The connection is open but a disconnect request has been made to the Transport Driver.
6	REOPEN:*	A connect request has been made on a connection that is in the CLOSING state.
7	FAIL:*	An error has been detected on the connection; the virtual circuit is not reliable.

Table 3-2. (Cont'd)

Value	State	Description
8	LISTEN:	An await connect call has been made for the socket but no connect request has been received.
9	FLUSH:*	All connect requests have failed; no virtual circuit has been established.
10	DIALURE:*	A dialect failure has occurred during VERIFY.
11	XWAIT:	The Transport Driver is waiting for the Resource Manager to complete the Transport Requester Block (see Section 4.1).

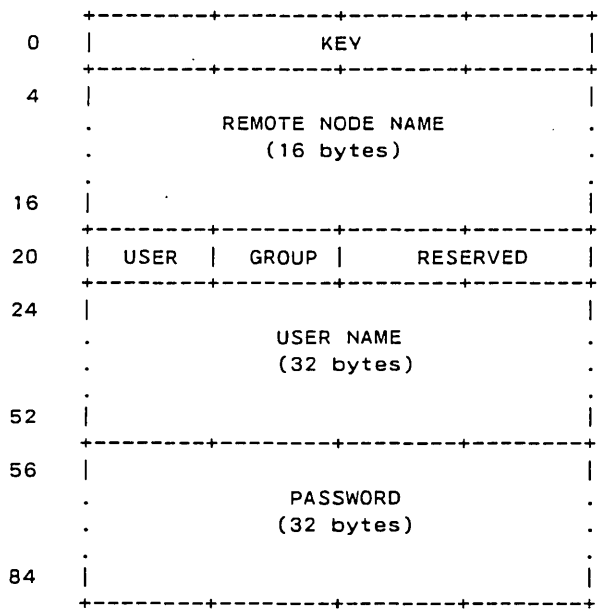
*These states occur only in connections between a FlexNet Network Resource Manager and Server Driver. They do not apply to connections established through the NET: device.

3.2 LOGON Table

The LOGON table contains information on remote accounts logged on. A separate LOGON table is available for each node logged on. Use LOOKUP to obtain the information, you cannot use GET. In the LOOKUP call, FlexNet interprets the name parameter as a node name. Node names are case sensitive and you can use wildcards.

You can also log on and off remote accounts using the LOGON table. The log on and off procedures are explained following the table description.

Note: FlexNet allows you to log on one account per node at a time. Logging on a second time automatically logs you off the account logged on first. PC DOS servers are different and allow you to log on multiple remote devices on the same node. See **FlexOS Versus PC DOS Servers** below for an explanation.



- **KEY:** Key field for LOOKUP
- **REMOTE NODE NAME:** Null-terminated remote node name
- **USER (R/O):** User ID on remote node for account specified in USER NAME
- **GROUP (R/O):** Group ID on remote node for account specified in USER NAME
- **USER NAME:** Account name on remote node
- **PASSWORD (Write-only):** Password required to log on the account or device specified in USER NAME. This field is used only with SET; the characters returned on LOOKUP are not valid.

FlexOS Versus PC DOS Servers

The FlexOS file security mechanism is based on user accounts and access privileges. The account is identified by a name--which specifies an entry in the computer's USER.TAB file--and indicates the account's user and group IDs. Users can log on one account at a time. If the user logs on another account on the same node, FlexOS automatically logs off the existing account.

The PC DOS file security mechanism is based on logical devices each with their own password. To access a device on a PC DOS server from a FlexOS requester, the device must have a predefined shortname. You cannot use the device's path specification from an FlexOS requester. In order to access two devices on the same node at the same time, the user logs on each one separately. Consequently, a user can be logged on a PC DOS server multiple times.

Logging On

Use the SET SVC and the LOGON table to begin a session between the FlexNet Network Resource Manager and Server Driver from within an application. Make the SET call with flag bit 0 set to 0 and fill in the LOGON table fields as follows:

- Put the remote node name in the REMOTE NODE NAME field.
- **For a FlexOS server:** Put the remote account name in USER NAME.
- **For a PC DOS server:** Put the remote shortname (the remote device's logical path name) in USER NAME.
- Put the account's (FlexOS) or the shortname's (PC DOS) password in the PASSWORD field.

All entries must be null-terminated. Do not enter node identifiers--for FlexOS, the succeeding :: and for PC DOS, the preceding \\--in the node names. Similarly, the trailing colon should be left off of device names. Note that you cannot specify a path to log on a PC DOS node; only predefined (see the PC DOS NET SHARE command) logical path names are valid. Leave all other LOGON fields null.

The SET call results in the following sequence of events and only returns success when all steps succeed.

- **Establish connection:** FlexNet selects a local and remote socket and attempts to establish a connection between these sockets.
- **Verify dialect:** FlexNet confirms the SMB dialect by sending a list of the dialects spoken and waiting for the remote node's selection.
- **Log on account:** FlexNet invokes the remote computer's log on procedure. On a FlexOS server, this procedure verifies the user account and password and returns the user and group IDs; FlexNet does not invoke the remote account's shell. On a PC DOS server, the procedure returns success after verifying that the device is sharable and the password is correct.

The return code from the SET call indicates whether or not FlexNet successfully logged on the node. Success is indicated by a zero return code; the error code indicates whether the operation failed on the connection, dialect verification, or log on.

FlexNet updates the LOGON table with the remote user and group IDs when the SET call is successful. For PC DOS servers, these fields are null. Note that user accounts can have different user and group IDs on different nodes.

Logging Off

You also use the SET command and LOGON table to end a requester/server session. To log off, set SET flag bit 0 to 1. You have the following log off options:

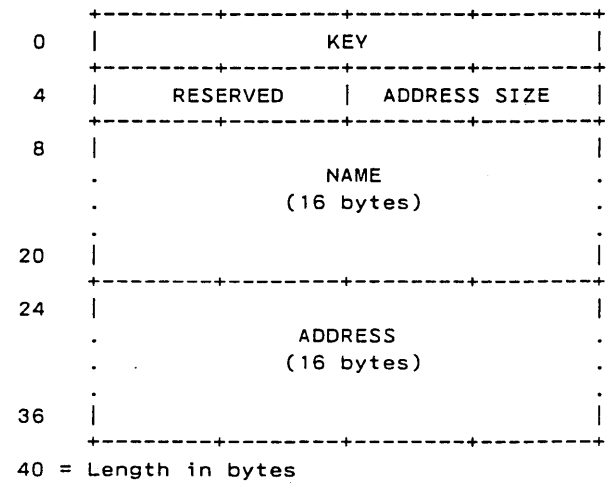
- To log off a single node, specify its name in the LOGON table's REMOTE NODE NAME field. If the node is a PC DOS server, this command logs off devices logged on.
- To log off a single device on a single PC DOS server, fill in the LOGON table's REMOTE NODE NAME and USER NAME fields with the device's node and shortname, respectively.
- To log off all nodes, specify a NULL remote node name.

3.3 NETNAME Table

Use the NETNAME table to retrieve and modify entries in the Name Server Driver's in-memory names database. Each entry indicates a node's or socket's physical network address. See Section 4.2 for the description of the FlexNet Name Server Driver.

Use LOOKUP to get the current entries; you cannot use GET. Specify the node or socket name in LOOKUP's name field and specify the name type in the LOOKUP SVC's flags field (name types are described in **Name Service Name Types** below). Node and socket names are case-sensitive and you can use wildcards.

Use SET to modify existing entries, delete existing entries, and add new entries. Changes made affect only the in-memory names cache--the disk-based, database file is not changed. Use SET's flags field to designate the name type.



- **KEY:** Key field for LOOKUP
- **ADDRESS SIZE:** Length in bytes of the address parameter
- **NAME:** Null-terminated node name
- **ADDRESS:** Node's physical network address

Name Service Name Types: Use the low order byte in the SET and LOOKUP flags field to designate the name type. Table 3-3 lists the valid field values.

Table 3-3. Name Service Name Types

SET Flag Bit	Value	Definition
7	0	Name is remote
	1	Name is local
6		Reserved and must be 0
4-5		Reserved for OEM use*
0-3	0	Local node name (LOCLNAME)
	1	Reserved
	2	Reserved
	3	Name of a remote node
	4	Name of a local or remote socket
	5-15	Reserved

*The OEM who wrote the Name Server Driver determines the purpose of these flag bits. If they are not implemented, set the flags to zero.

End of Section 3

Driver Interface

FlexNet requires custom driver development at two interfaces: the connection management and message transfer interface at the ISO transport level and the name service interface. These are the only FlexNet drivers that require program support; the remainder of the modules and drivers--the Network Resource Manager, NET: Device, Server Driver, and PC DOS server and requester drivers--are complete and do not require OEM customization. This section describes the driver functions you must support in your drivers.

Within each function, you can use the driver services described in the FlexOS System Guide and the SVCs described in the FlexOS Programmer's Guide. Recall, however, that you cannot issue an SVC from within a driver function that in turn calls one of the same driver's entry points. Also recall that an ASR cannot make an SVC that results in a process waiting. See the FlexOS System Guide for the complete set of rules governing the use of SVCs from within drivers, ASRs, and ISRs and the description of the calling protocol.

4.1 Transport Driver

To complete the computer's network interface, you must add the hardware and driver software that provide the transport, datalink, and physical level network services. FlexNet's interface to your driver is at the transport level and requires ISO level 4 8073-style virtual circuit services.

The Transport Driver consists of code and data groups as described in Section 4 of the FlexOS System Guide. The driver header in the data group provides entry points for a series of standard functions called by the driver's resource manager to control network I/O. All entry points can be called from both process and ASR contexts.

The parameter passed to each function is a pointer to a data structure in which information and buffer pointers are provided. Table 4-1 lists the driver entry points you must support and their purposes.

Note: The Network Resource Manager, the NET: Device Driver, Server Driver, and Name Server Driver all directly access the Transport Driver entry points. From the Transport Driver's point of view, however, it does not matter which of the higher level drivers acts as its resource manager. Consequently, the term "resource manager" is used generically in this section to refer to the Network Resource Manager and all other drivers that call the entry points.

Table 4-1. FlexNet Transport Driver Functions

Driver Function	FlexNet Operation	Description
INIT*		Initialization routine called by INSTALL to prepare for subsequent I/O. Return 62H as the Transport Driver type value and the type of subdriver required, if any.
SUBDRIVE*		Link routine called by INSTALL to establish a driver as a subdriver
UNINIT*		Uninitialization routine called by INSTALL to remove a driver
SELECT	CONNECT	Establish a connection to the specified node and socket (Opcode 01)
SELECT	AWAIT_CONNECT	Wait on a specified socket for a connection request (Opcode 02)
FLUSH	DISCONNECT	Terminate a connection (Opcode 05)
READ	RECEIVE	Wait for a message on a specific connection (Opcode 04)

*See the FlexOS System Guide for complete descriptions of the INIT, SUBDRIVE, and UNINIT functions.

Table 4-1. (Cont'd)

Driver Function	FlexNet Operation	Description
READ	DG_RECEIVE	Wait for a datagram (Opcode 08)
WRITE	SEND	Send a message over a specific connection (Opcode 03)
WRITE	DG_SEND	Send a datagram to a specific node and socket (Opcode 07)
GET		Get local node representation (Opcode 09)
SET		Set local node representation (Opcode 10)
SPECIAL	CANCEL	Cancel a message or datagram (Opcode 06)

4.1.1 Transport Request Block

The parameter passed to the Transport Driver by the resource manager is a pointer to a data structure referred to as the Transport Request Block (XRB). Figure 4-1 illustrates the Transport Request Block format; Listing 4-1 contains the C structure; and Table 4-2 contains the field descriptions.

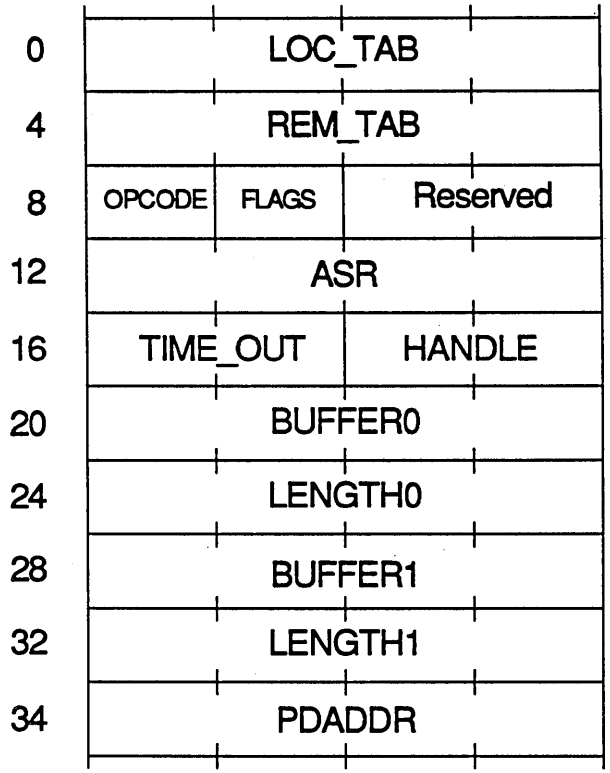


Figure 4-1. Transport Request Block

Listing 4-1. Transport Request Block C Definition

```
typedef struct
{
    TAB       *xrb_ltab;     /* loc_tab          */
    TAB       *xrb_rtab;     /* rem_tab          */
    BYTE       xrb_op;       /* opcode           */
    BYTE       xrb_flags;    /* flags            */
    UWORD      xrb_indx;     /* reserved         */
    VIOD     (*xrb_asr)();   /* asr              */
    UWORD      xrb_tmo;      /* timeout          */
    UWORD      xrb_handle;   /* handle           */
    BYTE      *xrb_buf0;     /* buffer 0         */
    ULONG      xrb_len0;     /* length 0         */
    BYTE      *xrb_buf1;     /* buffer 1         */
    ULONG      xrb_len1;     /* length 1         */
    BYTE      *xrb_pdaddr;   /* User process
                                descriptor addr.*/
}XRB;
```

Table 4-2. Transport Request Block Fields

Field	Description
LOC_TAB	Pointer to the Transport Address Buffer that specifies the local node and socket. This address is always in system space. Figure 4-2 illustrates the Transport Address Buffer.
REM_TAB	Pointer to the Transport Address Buffer that specifies the remote node and socket. This address is always in system space. Figure 4-2 illustrates the Transport Address Buffer.
OPCODE	The operation code of the requested driver function. The driver opcodes are listed in Table 4-1 above.
FLAGS	Bit map of flags Bit 0: 0 = BUFFER0 pointer is a system address 1 = BUFFER0 pointer is a user address Bit 1: 0 = BUFFER1 pointer is a system address 1 = BUFFER1 pointer is a user address Bits 2-7: Reserved
Reserved	Reserved for the Network Resource Manager; must not be used or modified by the Transport Driver.

Table 4-2. (Cont'd)

Field	Description
ASR	Address of the Asynchronous Service Routine. When provided in the XRB, schedule this ASR after completing the function (see Section 4.1.2 below for the doasr call description).
TIMEOUT	Time limit for the operation which, if exceeded, should result in a timeout error. The value indicates 100 millisecond increments; a zero value indicates no time limit.
HANDLE	Connection number of the virtual circuit linking specific local and remote node and socket pairs
BUFFER0	Pointer to the first message buffer
LENGTH0	Length in bytes of the first message buffer
BUFFER1	Pointer to the second message buffer
LENGTH1	Length in bytes of the second message buffer
PDADDR	Process descriptor address of process that owns the buffers. If a buffer is in user space, this is the address you use in your MAPU call. However, this is not necessarily the process calling this entry point and therefore not the PDADDR you use in your FLAGSET calls. Get the PDADDR for FLAGSET calls from the Driver Header's Ready List Root (RLR). This field is only valid if user buffers are indicated.

The Transport Driver's resource manager provides the function's local and remote node and socket addresses in a data structure called the Transport Address Buffer. Figure 4-2 illustrates its format.

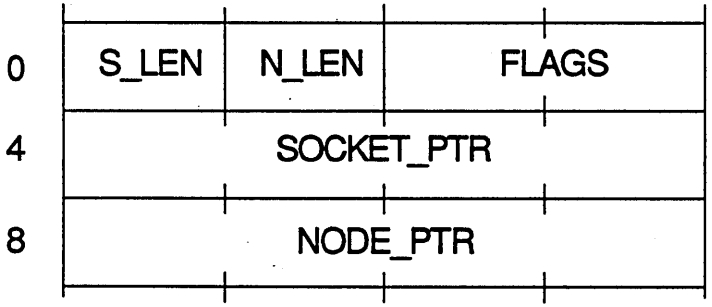


Figure 4-2. Transport Address Buffer

The S_LEN and N_LEN values indicate the length in bytes of the buffers for the socket and node representations, respectively.

FLAGS field bit 0 is the only significant bit in the word. When the bit is set, it means you must pick the socket. When it is zero, the socket is provided.

SOCKET_PTR is the pointer to the socket representation buffer and NODE_PTR is the pointer to the node representation buffer. Both addresses are in system space. The form of the node and socket representation can be an address or a name depending on the Name Server Driver and the requirements of the network hardware. "address" is used in the descriptions that follow to refer to both forms.

4.1.2 Transport Driver Functions

The calling resource manager builds the Transport Request Block (XRB) and provides the pointer when it calls the Transport Driver entry point. Note that not all fields in the XRB are filled in for all calls. The function descriptions below indicate which fields are valid, what the function is expected to do, and what it is required to provide in return.

Note: All transport functions for which a timeout value is must provide their own time out mechanism.

All entry points except CANCEL, GET, and SET must be able to support asynchronous calls. You must return two codes when the

asynchronous form is used--one from the driver's synchronous portion and the other from its asynchronous portion.

The synchronous portion of the driver should return 0 to indicate success or an error code to indicate failure. The Transport Driver return codes are listed in Appendix C. Do not return an event mask; the user process gets its emask value from the Transport Driver's resource manager when the success code is received. When you return success, the resource manager expects to receive the request's completion code through the ASR it provided to the function. If you return an error code, the resource manager does not wait for the ASR--it passes along the error code as the completion code.

To return the request's completion code from the asynchronous driver code, schedule the ASR provided as follows:

```
doasr (asr_routine,ret,&xrb,priority)
```

Use the function's return code for the ret argument. Your CONNECT and AWAIT_CONNECT routines should return the handle as the ret value to indicate success. Use the pointer to the Transport Request Block as the doasr parm2 value. The other entry points should return 0 to indicate success. Set the ASR priority at 200.

The function descriptions are presented in opcode order. Each description shows the Transport Request Block (XRB) parameters provided and an explanation of the function's purpose.

SELECT: CONNECT--Establish a connection

XRB Parameters Provided:

LOC_TAB	Pointer to local node Transport Address Buffer. The local node address is provided but the local socket address is only provided if FLAGS bit 0 is zero. If FLAGS bit 0 is set, you must pick a local socket.
REM_TAB	Pointer to remote node Transport Address Buffer. Both node and socket addresses are provided.
OPCODE	01
ASR	Address of the Asynchronous Service Routine
TIME_OUT	Number of 100 millisecond increments before timeout

Return Code:

XP_SUCCESS	Operation was successful (synchronous portion of driver only)
handle	Connection number for virtual circuit (asynchronous portion of driver only)
error	Error code

The Transport Driver's resource manager calls the SELECT entry point for two purposes: 1) to issue a connection request to another node and 2) to wait for a connection request from another node. When the OPCODE value is 01, the resource manager is calling to issue a connection request.

The resource manager provides the remote node and socket addresses in the REM_TAB buffers and the local node address in the LOC_TAB NODE buffer. The local socket address is a programmer-option.

Check FLAGS bit 0 to determine if a local socket is provided. When the value is 0, the socket address is at the SOCKET_PTR and its address length is indicated by S_LEN. When the value is 1, there's no socket in the buffer and S_LEN indicates the maximum buffer length.

Before returning, select a socket, record its address at the SOCKET_PTR buffer, and change S_LEN to indicate the actual length.

After the connection has been established, assign the handle value return it in two places:

- Put the number in the HANDLE field in the XRB
- Use it as the ret value in the doasr parm1 argument.

Note that the handle value does not have to be the same on the local and remote nodes.

SELECT: AWAIT_CONNECT--Wait for a connection request

XRB Parameters Provided:

LOC_TAB	Pointer to local node's Transport Address Buffer.
REM_TAB	Pointer to remote node's Transport Address Buffer
OPCODE	02
ASR	Address of the Asynchronous Service Routine.
TIME_OUT	Number of 100 millisecond increments before timeout

Return Code:

E_SUCCESS	Operation was successful (synchronous portion of driver only)
handle	Connection number for virtual circuit (asynchronous portion of driver only)
error	Error code

The resource manager calls the SELECT entry point with OPCODE 02 to wait for a connection request. LOC_TAB contains the local node and socket addresses. Use the REM_TAB buffers to return the remote node and socket addresses to the calling process. The REM_TAB S_LEN and N_LEN values indicate the maximum buffer lengths. Change these to indicate the actual address lengths before returning.

After the connection has been established, define the connection handle before calling the ASR. You return the handle value in two places:

- Put the number in the HANDLE field in the XRB
- Use it as the ret value in the doasr parm1 argument.

Note that the handle value does not have to be the same on the local and remote nodes.

WRITE: SEND--Send Message on an Established Connection

XRB Parameters Provided:

ASR	Address of the Asynchronous Service Routine
OPCODE	03
FLAGS	Bit 0: 0 = BUFFER0 pointer is a system address 1 = BUFFER0 pointer is a user address Bit 1: 0 = BUFFER1 pointer is a system address 1 = BUFFER1 pointer is a user address
TIME_OUT	Number of 100 millisecond increments before timeout
HANDLE	Connection number of virtual circuit
BUFFER0	Pointer to first buffer to send
LENGTH0	Length in bytes of first buffer
BUFFER1	Pointer to second buffer to send
LENGTH1	Length in bytes of second buffer
PDADDR	Process descriptor address of the process that owns the buffers

Return Code:

E_SUCCESS	Operation was successful (synchronous and asynchronous portions of the driver)
error	Error code

The resource manager calls the WRITE entry point to send a message on an established connection or to send a datagram. When the OPCODE is 03, the first option is selected (see WRITE: DG_SEND for the description of the second option). This option requires you to guarantee that the message has been successfully received by the other node before scheduling the ASR.

The resource manager indicates which connection to use with the handle value and provides the message to be sent in two buffers.

BUFFER0 contains the first part of the message and BUFFER1 the second part. If the length value for either buffer is NULL, do not send that buffer.

Be sure to check the flag bit value for each buffer--the asynchronous portion of the driver must call the MAPU and SADDR driver services to convert a buffer in user space to system space before it can access its contents. (The synchronous portion of the driver runs in process context and, hence, does not require address conversion.) Note that the flag value of each buffer can be different.

READ: RECEIVE--Receive Message on an Established Connection

XRB Parameters Provided:

ASR	Address of the Asynchronous Service Routine
OPCODE	04
FLAGS	Bit 0: 0 = BUFFER0 pointer is a system address 1 = BUFFER0 pointer is a user address
TIME_OUT	Number of 100 millisecond increments before timeout
HANDLE	Connection number of virtual circuit
BUFFER0	Pointer to receive buffer
LENGTH0	Length in bytes of BUFFER0. Return the number of bytes actually received.
PDADDR	Process descriptor address of the process that owns the buffer

Return Code:

E_SUCCESS	Operation was successful (synchronous and asynchronous portions of the driver)
error	Error code

The resource manager calls the READ entry point to receive a message on an established connection or to receive a datagram. When the OPCODE is 04, the first option is selected (see READ: DG_RECEIVE for the description of the second option). This option requires you to guarantee the integrity of the data received in the message before scheduling the ASR provided.

The resource manager provides a pointer to the buffer where you put the message, the length of the buffer, and the connection handle indicating which virtual circuit to wait on. If the entire message is too large to fit in the buffer, return the E_MOREDATA error code in the doasr parm1 argument. On the next RECEIVE call for that handle, put the remainder of the message or the next block of data in the buffer. Repeat the E_MOREDATA error if the remainder of the message does not fit.

Return the E_SUCCESS when the message or the remainder of the message fits in the buffer. Before returning, update the LENGTH0 field to indicate the actual number of data bytes put in the buffer.

Note: Do not return more than one message in the buffer with a single RECEIVE call.

FLUSH: DISCONNECT--Close a Virtual Circuit

XRB Parameters Provided:

ASR	Address of the Asynchronous Service Routine
OPCODE	05
TIME_OUT	Number of 100 millisecond increments before timeout
HANDLE	Connection number of virtual circuit to close

Return Code:

E_SUCCESS	Operation was successful (synchronous and asynchronous portions of the driver)
error	Error code

The resource manager calls FLUSH to close the virtual circuit indicated by the handle. Subsequent attempts to use the specified handle should return the E_BADHANDLE error code. You should return the E_BADCONN error code for all requests pending when FLUSH is called. However, wait for all outstanding sends to complete before returning.

Note: If a network error prevents FLUSH from completing, disassociate the handle at the local end and return the E_REQTMO error code instead of the E_NETERR value.

SPECIAL: CANCEL--Abort a Request

XRB Parameters Provided:

OPCODE	06
BUFFER0	Pointer to XRB to cancel

Return Code:

E_SUCCESS	Operation was successful
error	Error code

The resource manager calls CANCEL to abort a request. The request is designated by the Transport Request Block at the address provided in the BUFFER0 field. This address is always in system space. If that request has already been completed return the E_REQDONE error code. If the event has not already completed, cancel it and return the E_CANCELLED error code for that request.

WRITE: DG_SEND--Send a datagram

XRB Parameters Provided:

LOC_TAB	Pointer to the local node's Transport Address Buffer
REM_TAB	Pointer to the remote node's Transport Address Buffer
OPCODE	07
FLAGS	Bit 0: 0 = BUFFER0 pointer is a system address 1 = BUFFER0 pointer is a user address Bit 1: 0 = BUFFER1 pointer is a system address 1 = BUFFER1 pointer is a user address
ASR	Address of the Asynchronous Service Routine
BUFFER0	Pointer to first buffer to send
LENGTH0	Length in bytes of first buffer
BUFFER1	Pointer to second buffer to send
LENGTH1	Length in bytes of second buffer
PDADDR	Process descriptor address of the process that owns the buffers

Return Code:

E_SUCCESS	Operation was successful (synchronous and asynchronous portions of the driver)
error	Error code

The resource manager calls the WRITE entry point with OPCODE 07 to send a datagram. DG_SEND does not require an open connection and it does not need to guarantee reception. Get the destination node and socket addresses from the REM_TAB buffer. The LOC_TAB specifies the local node and socket addresses you should use.

The message to send is in buffers BUFFER0 and BUFFER1; BUFFER0 contains the first part of the message and BUFFER1 the second. If the length for either buffer is NULL, do not send the buffer.

Be sure to check the flag bit value for each buffer--the asynchronous portion of the driver must call the MAPU and SADDR driver services to convert a buffer in user space to system space before it can access its contents. (The synchronous portion of the driver runs in process context and, hence, does not require address conversion.) Note that the flag value of each buffer can be different.

READ: DG_RECEIVE--Receive a datagram

XRB Parameters Provided:

LOC_TAB	Pointer to the local node's Transport Address Buffer
REM_TAB	Pointer to the remote node's Transport Address Buffer
OPCODE	08
FLAGS	Bit 0: 0 = BUFFER0 pointer is a system address 1 = BUFFER0 pointer is a user address
ASR	Address of the Asynchronous Service Routine
TIME_OUT	Number of 100 millisecond increments before timeout
BUFFER0	Pointer to receive buffer
LENGTH0	Length in bytes of BUFFER0. Return the number of bytes actually received.
PDADDR	Process descriptor address of the process that owns the buffer

Return Code:

E_SUCCESS	Operation was successful (synchronous and asynchronous portions of the driver)
error	Error code

The resource manager calls the READ entry point with OPCODE 08 to

receive a datagram. DG_RECEIVE does not require an open connection. The LOC_TAB provides the node address and the socket to listen on. The REM_TAB contains buffer pointers and maximum length values.

The BUFFER0 pointer indicates the address of the receive buffer. Be sure to check FLAGS flag bit 0 to determine if the buffer is in user or system space.

Before returning, put the incoming message's source node and socket addresses in the corresponding REM_TAB buffers and set the S_LEN and N_LEN values to indicate the address lengths. Set the LENGTH0 value to indicate the number of data bytes received.

Note: No more than one message should ever be returned in the buffer with a single call.

GET: Get local ID

XRB Parameters Provided:

LOC_TAB	Pointer to the local node's Transport Address Buffer.
OPCODE	09

Return Code:

E_SUCCESS	Operation was successful
error	Error code

The resource manager calls the GET entry point to determine the local node address. Fill in the LOC_TAB structure's N_LEN value and the node address at the NODE_PTR address provided before returning.

SET: Set Local ID

XRB Parameters Provided:

LOC_TAB	Pointer to the local node's Transport Address Buffer.
OPCODE	10

Return Code:

E_SUCCESS	Operation was successful
error	Error code

The resource manager calls the SET entry point to set the local node name. The only valid data in the LOC_TAB provided are the N_LEN value and the node name at the NODE_PTR address. This call provides you with the node's logical name string, not the address. You should verify that the name is unique on the network before returning. If it is not, return the E_INUSE error code.

This entry point is called by the Name Server's SET function. The OEM must determine how the Transport Driver and Name Server Driver work together to estblish local node names.

A successful set of the local node name can be allowed only once. Subsequent attempts should return the E_LNAMESET error code.

4.2 Name Server Driver

The Name Server Driver is responsible for translating between node names and addresses. It is called by the Network Resource Manager and NET: Device Driver.

The FlexNet name service has three components: the Name Server Driver, the NAMES utility, and the disk-based NETNAMES.DAT names database file. This section describes the names reserved for FlexNet use, the Name Server Driver functions you must support, the NETNAMES.DAT file, and the remote node translation feature. See the FlexNet User's Guide for the NAMES utility description.

The FlexNet Name Server Driver provides the fundamental name translation services necessary to get the network running. Use it as the basis for constructing your own driver.

4.2.1 Reserved Names

The names listed in Table 4-3 are reserved for system use.

Table 4-3. FlexNet Reserved Names

Name	Description
DRINAMES	Name server socket name used to issue and receive remote requests for name service
DRIRQSTR	Requester socket name used by Network Resource Manager to connect with FlexNet servers.
DRISRVR	Server socket name used by Server Driver to connect with FlexNet requesters.
LOCLNAME	Local node's logical name

Set the addresses for the DRINAMES, DRIRQSTR, and DRISRVR sockets in the synchronous portion of your Name Server Driver. Each socket

must have the same address on all nodes. These names are always uppercase.

DRIRQSTR and DRISRVR are explained in Section 1.2. DRINAMES is reserved for remote name translations; see Section 4.2.5 below for the description of its use.

LOCLNAME is every node's logical name until the actual name is set by the NAMES command. The network cannot be used until LOCLNAME has been redefined.

4.2.2 Name Server Driver Data Structures

Two data structures are provided to the Name Server Driver. The Network Resource Manager provides a pointer to the Shared Data Structure when it calls the Name Server Driver's SELECT entry point. The Network Resource Manager and NET: Device Driver provide a pointer to the Name Server Element when they call the GET and SET entry points. For all calls, the address provided is in system space.

Shared Data Structure

Figure 4-3 illustrates the Shared Data Structure. Listing 4-2 contains the C structure definition. The explanation of the contents follows the listing.

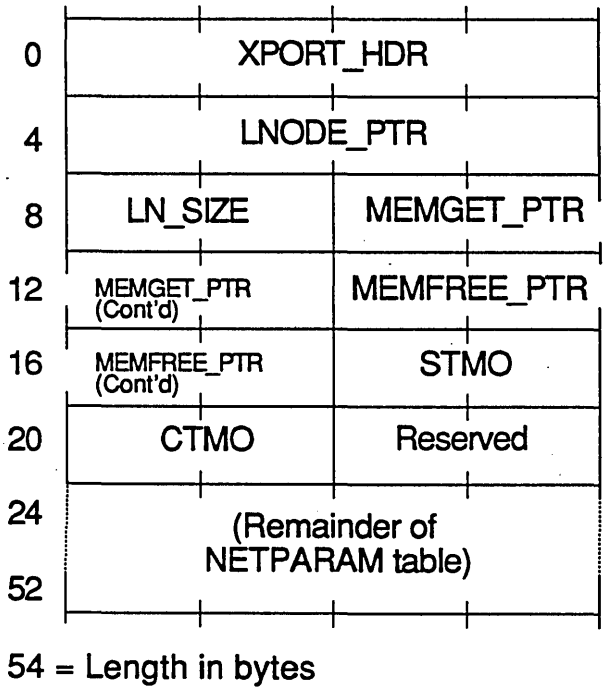


Figure 4-3. Shared Data Structure

```
struct sds
{
    DH        *nt_xport;       /* XPORT_HDR: Transport Drvr */
    BYTE      *nt_local;       /* LNODE_PTR: Local address  */
    UWORD     *nt_lsiz;        /* LN_SIZE: Address size     */
    BYTE      *(*nt_mget)();   /* MEMGET_PTR: Get mem. block*/
    VOID      (*nt_mfree)();   /* MEMFREE_PTR: Release mem.
                                  block                     */
    NETPARAM nt_np;            /* STMO, CTMO, ...: NETPARAM
                                  table values              */
};
```

Listing 4-2. Shared Data Structure C Definition

The Shared Data Structure fields are defined as follows:

- **XPORT_HDR:** Pointer to the Transport Driver header. See the FlexOS System Guide for the driver header format and DH definition. This provides you with access to the network for getting and setting the local address, remote node name translations, and other name service functions that require network access.
- **LNODE_PTR:** Pointer to a buffer with the local node address.
- **LN_SIZE:** Size in bytes of the local node address.
- **MEMGET_PTR:** Pointer to a routine that allocates up to and including 256 bytes from the Network Internal Memory Pool (NIMP). Call syntax is

  ```
  addrs = (*sds_ptr->memget_ptr)(nbytes);
  ```

  where sds_ptr is the Shared Data Structure pointer, nbytes is a WORD value indicating the number of bytes required, and addrs is the address of the memory block.
- **MEMFREE_PTR:** Pointer to a routine that releases NIMP memory allocation. Call syntax is

  ```
  (*sds_ptr->memfree_ptr)(addrs)
  ```

  where sds_ptr is the Shared Data Structure pointer and addrs is the memory block address returned from the MEMGET call.
- **STMO, CTMO, ... :** Network parameter values. The network parameters are organized as shown in the NETPARAM table. See Section 5.1 for the NETPARAM table description.

The Network Internal Memory Pool introduced above is one of the network parameters. See the NETPARAM table description for the explanation of its purpose and how you set its size.

Name Server Element

Figure 4-4 illustrates the Name Server Element. Listing 4-3 provides the C structure definition.

The Name Server Element fields are defined as follows:

- **NAME_PTR:** Pointer to the node or socket name buffer. For address requests and name deletes, the name can contain wildcard characters *, ? and/or ^. For name requests, pointer indicates location where you put the name. The name is always NULL terminated.
- **ADDR_PTR:** Pointer to the address buffer. When NULL, delete the name specified from the memory-resident names database.

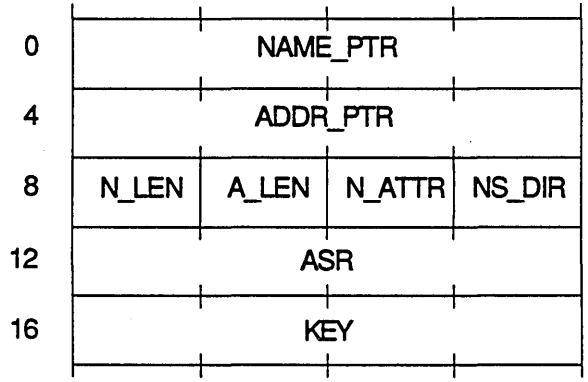


Figure 4-4. Name Server Element Format

- **N_LEN:** Length in bytes of the name buffer. Value indicates either the actual length (when the name is supplied) or the maximum length (when the address is supplied). Change N_LEN to the actual length when you provide the name.
- **A_LEN:** Length in bytes of the address buffer. Value indicates either the actual length (when the address is supplied) or the maximum length (when the name is supplied). Change A_LEN to the actual length when you provide the address.

- **N_ATTR:** Bit map indicating type of name as follows:

  Bit 7: 1 = Local name
  0 = Remote name
  Bit 6: Reserved for system use
  Bits 4–5: OEM-defined
  Bits 0–3: Evaluate as a group according to the following values:
  0 = Name is LOCLNAME
  1 Reserved
  2 Reserved
  3 = NSE of a node
  4 = NSE of a socket
  5 – 15 Reserved

- **NS_DIR:** Value indicating direction of name search as follows:

  - 00--Address request: Provide the address for the node name given.
  - 01--Name request: Provide the name for the node address given.

  Be sure to update N_LEN or A_LEN when you provide the name or address.

- **ASR:** Address of the ASR you schedule if the asynchronous portion of the driver must be accessed.

- **KEY:** Key value of name or address provided. NULL indicates you search from the beginning. A value is an index into the table indicating the item from which you begin the search. Assign the key value for all nodes in the local name service database and return the index on all GET calls.

Listing 4-3. Name Server Element (NSE) C Definition

```
typedef struct
{
    BYTE    *name_ptr;    /* Pointer to name string        */
    BYTE    *addr_ptr;    /* Pointer to address string     */
    BYTE     n_len;       /* Length of name string         */
    BYTE     a_len;       /* Length of address string      */
    BYTE     n_attr;      /* Bit map indicating name type */
    BYTE     ns_dir;      /* Direction of name search      */
    VOID     asr          /* ASR address                   */
    LONG     key          /* Key value of name             */
}NSE;
```

4.2.3 Name Server Driver Functions

You must support the Name Server Driver INIT, UNINIT, SELECT, GET, and SET entry points. The remainder of the entry points are not called.

INIT

FlexOS (through the INSTALL SVC) calls INIT to initialize the port and return the driver type value. No parameters (that is, no INSTALL flags or unit number) are provided with this call. Return 65H as the Name Server Driver type value in your return code.

UNINIT

The Network Resource Manager calls the UNINIT entry point to remove the driver. Release any memory used, clear flags, and terminate processes created by the Name Server Driver before returning.

SELECT

The Network Resource Manager calls SELECT once the network parameters are set and passes a pointer to the Shared Data Structure. Save this address. You need the address of the Transport Driver header and the NNTE value to perform the GET and SET functions. The values in the Shared Data Structure are not valid until the local name is set.

GET--Return Name Database Information

NSE Parameters Provided:

NAME_PTR	Name buffer pointer
ADDR_PTR	Address buffer pointer
N_LEN	Length in bytes of name buffer
A_LEN	Length in bytes of address buffer
N_ATTR	Type of element
NS_DIR	Search direction
ASR	Address of Asynchronous Service Routine
KEY	Index into local database of entry

Return Code:

E_SUCCESS	Operation was successful and completed by synchronous portion of the driver.
E_ASYNC	Synchronous portion of the driver could not complete the name search (that is, the name is not memory-resident) and asynchronous portion of driver was called. This is not an event mask.
error	Error code

The resource manager calls GET to obtain information on an entry in the name service database. The parameter provided is a pointer to a Name Server Element. Check the memory-resident entries in the synchronous portion of the driver. If the name is found, update the Name Server Element buffers and fields and return success. The socket names DRIRQSTR, DRISRVR, and DRINAMES must be translated in the synchronous portion of the driver.

Call the Transport Driver GET entry point when the Name Server Element N_ATTR byte indicates a request for the LOCLNAME if the local node address is not already in the database. You must pass the Transport Driver a pointer to a Transport Request Block with OPCODE 09 and a pointer to a Transport Address Buffer in the LOC_TAB field. Set the N_LEN value to the maximum address size; S_LEN is irrelevant. You get back the address in the Transport Address Buffer's NODE_PTR buffer. Fill in the Name Server Element with this information before returning.

If the information is not available in the memory-resident names database, return the E_ASYNC return code in the driver's synchronous portion and check the NETNAMES.DAT disk file; Section 4.2.4 describes the file format. After you have filled in the appropriate NSE fields, schedule the ASR provided as follows:

```
doasr(asr_routine,retcode,&nse,prior)
```

The doasr parm1 argument is the GET call's return code (E_SUCCESS or one of the error codes); the second argument is the pointer to the completed NSE.

The Name Server Driver is responsible for reserving the memory and managing the cache of name entries from the NETNAMES.DAT file. The number of entries to be stored in the cache is specified in the NNTE parameter in the NETPARAM table. You get this value from the Shared Data Structure. Be sure to check the Name Server Element's a KEY value for an index into the cache and to provide the index when you return entries from it.

Note: Section 4.2.5 describes how to use the DRINAMES socket and the network to translate a node name not available from the memory or disk-based names databases.

SET--Add or Delete a Memory-resident Entry

NSE Parameters Provided:

NAME_PTR	Name buffer pointer
ADDR_PTR	Address buffer pointer
N_LEN	Length in bytes of name buffer
A_LEN	Length in bytes of address buffer
N_ATTR	Type of element

Return Code:

E_SUCCESS	Operation was successful
error	Error code

The resource manager calls SET to add or delete an entry in the memory-resident names database. You add the name when both name and address strings are provided. You delete the name when the name

is provided but the address string is a NULL pointer. Under no circumstances should the SET function modify NETNAMES.DAT.

A SET call with the Name Server Element N_ATTRIB value indicating the LOCLNAME is a special case. It indicates that the user has finished making all network parameter modifications and is ready to use the network.

In response to a SET LOCLNAME call, make sure that this is the first time the local name has been set and, if it is, prepare a Transport Request Block (XRB) with OPCODE 10 and the local name in the LOC_TAB structure's NODE buffer. Call the Transport Driver SET entry point and pass the pointer to the XRB in the call. A success return code indicates that the node name is unique.

Finish the initialization of the Name Server Driver when the Transport Driver returns success. This would include getting the NNTE value from the Shared Data Structure and allocating sufficient memory to store the names cache, getting any flags required for the asynchronous portion of the driver, creating processes, and other housekeeping chores.

4.2.4 Name Database File Components

The name database file is an ASCII file consisting of a series of entries. You create the file with an editor. The file must be named NETNAMES.DAT and be recorded on the system: disk. Table 4-4 lists the components of each entry. Figure 4-5 illustrates the NETNAMES.DAT entry format.

```
NAME:        ACCOUNTS
ADDRESS:     "ACCOUNTS"
TYPE:        NODE
/*Accounting office, Hank's systems  */

NAME:        PERSONNEL
ADDRESS:     30,31,,32,33,34
/* This is address "01n234" where */
/* n = hex 00 */
TYPE:        REMOTE SOCKET
```

Figure 4-5. Sample NETNAMES.DAT Entries

Table 4-4. Name Database Fields

Field	Description
NAME:	Logical node or socket name. Node names can be up to 15 alphanumeric characters long; socket names up to eight. For both, the first character must be alphabetic.
ADDRESS:	Corresponding physical address. The address can be up to 16 characters long and in one of two formats: **string:** Address consists of alphanumeric characters enclosed by quotation marks; for example, "Address5". Entries are case-sensitive. **hexadecimal:** Address consists of the hexadecimal byte value of each ASCII character. Separate each byte with a comma.
TYPE:	Entry type. Fill in with one of the following: **NODE** indicates the entry is a node. **LOCAL SOCKET** indicates the entry is a local socket. **REMOTE SOCKET** indicates the entry is a remote socket.
OEM:	Implementation-dependent field not used by the NAMES utility.
/* ... */	Comments field. You can put comments on any line.

4.2.5 Remote Name Translation

Remote node translation is a Name Server Driver GET function that uses datagrams through the DRINAMES socket to find the address for a node name not in the local memory or NETNAMES.DAT databases. Figure 4-6 illustrates the datagram message format used for this function.

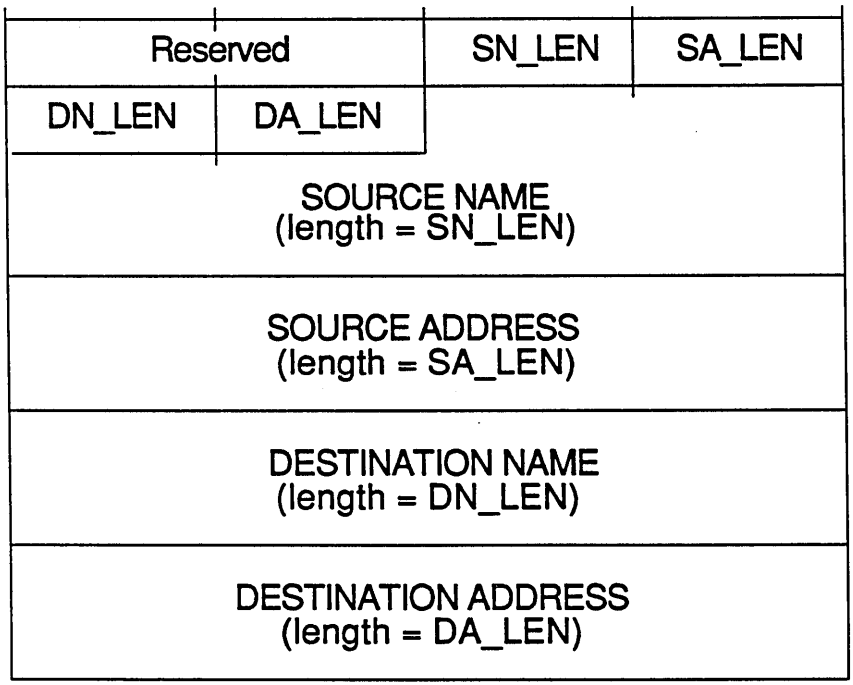


Figure 4-6. Remote Node Translation Datagram Format

SOURCE NAME and ADDRESS are the name and address of the node making the request. DESTINATION NAME is the name of the node for which the address is unknown. The four _LEN values indicate how long the corresponding name and address fields are. For translation requests, the DA_LEN value is NULL and there is no destination address. In the response message, all fields are filled.

To send a request, the Name Server Driver makes a Transport Request Block for a DG_SEND (OPCODE 07). The fields that must be completed are:

- LOC_TAB: Fill in the local node and DRINAMES addresses.
- REM_TAB: Fill in the node and DRINAMES addresses.
- FLAGS: Set to 0
- ASR: Fill in address of Asynchronous Service Routine
- BUFFER0: Provide pointer to buffer.with the above message
- LENGTH0: Fill in with message length

All other fields are null.

Make the send call from the asynchronous portion of the driver. Call the Transport Driver WRITE entry point (you get the address from the Shared Data Structure) and pass the Transport Request Block address.

To receive a request, the Name Server Driver makes a DG_RECEIVE (OPCODE 08) call to the Transport Driver. The Transport Request Block fields that must be filled in are

- LOC_TAB: Fill in the local node and DRINAMES addresses
- REM_TAB: Provide pointers to node and socket address buffers
- FLAGS: Set to 0.
- ASR: Fill in address of Asynchronous Service Routine
- TIME_OUT: Set to 0.
- BUFFER0: Provide pointer to receive buffer.
- LENGTH0: Set to maximum length of message.

Make the receive call from the asynchronous portion of the driver. Call the Transport Driver READ entry point (you get the address from the Shared Data Structure) and pass the Transport Request Block address.

When a message is received, determine if it is a request (DA_LEN equals 0) or a response (DA_LEN is not 0). For requests, check your names database to see whether or not you have an address for that name. If you do not, do not respond. If you do, complete the message's DESTINATION ADDRESS and DA_LEN fields and send the datagram back to the source node. For responses, add the name to your memory-resident names cache.

End of Section 4

FlexNet Configuration Options

Several aspects of FlexNet operation are governed through a set of parameters. Default values are provided for these parameters. However, you are encouraged to modify them to accommodate your hardware requirements, improve performance, reduce memory overhead, and so forth. The parameters fall into the following broad categories.

- **Timeouts:** Default timeout period values for a send message, connection attempt, and broken connection
- **Transaction Limits:** Maximum values for connections, operations in progress, log ons for servers and requesters, server processes, and entries in memory-resident name service database
- **Memory requirements:** Number and size of the large buffers and the FlexNet internal memory pool
- **Miscellaneous:** Server security mode (normal or extended), watchdog ping granularity, and password encryption key

Note: Timeout values apply to requester/server transactions between the Network Resource Manager and the Server Driver only, not between local and remote NET: devices. However, the transaction limits apply to communication between local and remote NET: devices as well as Network Resource Manager to Server Driver connections.

The FlexNet operating parameters are available through the NETPARAM table. You set these values with the SET SVC or the NETSET utility. The remainder of this section describes the NETPARAM table parameters and how to set them.

5.1 NETPARAM Table

The NETPARAM table contains the FlexNet operating parameters you can modify. Use the GET SVC to find out the current values and the SET SVC to change them. Alternatively, values can be set with the

NETSET utility. You must make all changes before you set the node's local name with the NAMES command. An attempt to change NETPARAM values either with SET or NETSET after the local name has been established returns an error message. The FlexNet User's Guide contains a sample CONFIG.BAT file that illustrates NAMES and NETSET use.

The NETPARAM table (table number 60H) is maintained by the Network Resource Manager. It is available for modification before any FlexNet drivers have been selected. The following figure illustrates the format of the NETPARAM table. Table 5-1 summarizes each field's purpose.

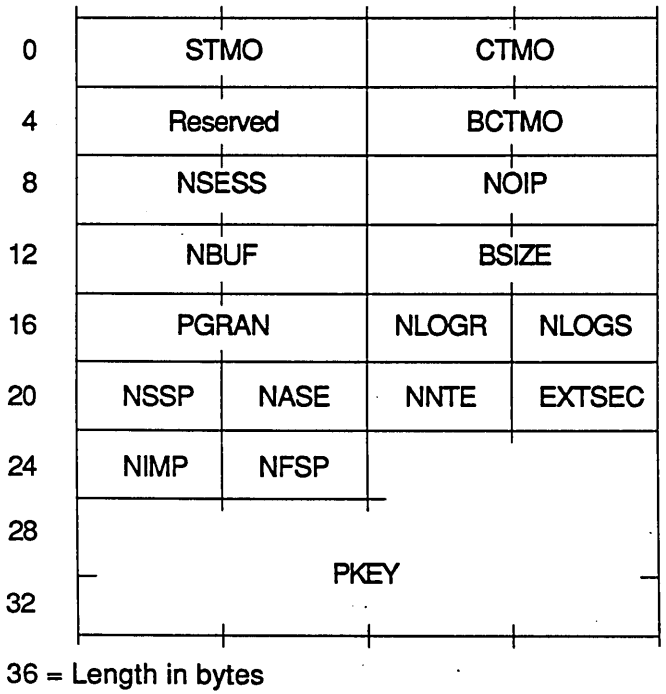


Figure 5-1. NETPARAM Table

Table 5-1. NETPARAM Table Field Definitions

Parameter	Default	Definition
STMO	5	Default send timeout value in seconds. Set to zero to disable timeout.
CTMO	30	Default connection timeout in seconds. Set to zero to disable timeout.
BCTMO	3	Broken connection timeout. The Server Driver accepts a limited number of missing echo messages before assuming a connection is broken. This value represents the number of echos messages that must be received. Use the default as the minimum value. Set to zero to disable timeout.
NSESS	16	Maximum number of connections. The minimum is two.
NOIP	32	Maximum simultaneous network operations in progress allowed per requester process. The range is 4 through 256.
NBUF	12	Number of large buffers. Large buffers are used for all network messages longer than 256 bytes. The minimum number of large buffers is four.
BSIZE	4	Kilobytes in each large buffer. Maximum value is 64.

Table 5-1. (Cont'd)

Parameter	Default	Definition
PGRAN	30	Frequency, in seconds, of echo message issued by requester. Server Driver listens for echo message to confirm connection is not broken.
NLOGR		Number of remote log-ons allowed per process family on a requester; zero specifies no limit.
NLOGS		Number of remote log-ons allowed per server; zero specifies no limit.
NSSP	1	Number of synchronous server processes. Server nodes must have at least one.
NASE	24	Number of asynchronous server events. Server nodes must have at least one; 24 is the maximum.
NNTE	32	Maximum number of name table entries to be stored in memory.
EXTSEC	0	Server Driver's security mode. Bit 0 selects extended security when set (1) or normal security when off (0). Bits 1 through 7 are not used and must be zero.
NIMP	16	Size in kilobytes of internal memory pool.
NFSP	0	Number of fast synchronous server processes. An FSP services FlexOS calls that do not result in excessive server delay, such as Read, Write, and Special.
PKEY		10-byte password encryption key.

Setting the NETPARAM table parameters is an installation function that draws its values from the network configuration and the node's expected use. For several parameters, trial and error is the best technique for optimizing a value. Each parameter's considerations and consequences are described below. The presentation is organized by the general categories defined above.

5.1.1 Selecting Timeout Values

The timeout values determine how long you wait for a specific transaction to complete before assuming an error has occurred. The goal is to specify a realistic value that allows for an occasional delay (caused, for example, by heavy network traffic) without waiting unnecessarily for an event that is never going to happen.

STMO

The send timeout value determines how long the Network Resource Manager and Server Driver wait for the acknowledge message from a SEND call. To arrive at a reasonable value, add the worst case time limits for all modules involved with message handling. The contributing modules include the local and remote Transport Drivers, the network itself, the remote Server Driver, and so forth. Increase the STMO if you get E_REQTMO errors frequently.

CTMO

The connection timeout specifies how long the Network Resource Manager waits for a connection request to be processed. Besides the time factors described in STMO, there are two other contributors to delays on connection attempts: all of the server's virtual circuits are allocated or the server is busy servicing other requests. Regarding the first factor, the connection cannot be established until another connection is terminated. Regarding the second, the connection can be established as soon as an asynchronous event becomes available.

BCTMO

The BCTMO is the number of the echo message in a sequence of missing echo messages the Server Driver must receive to keep a connection established. For example, a BCTMO of 3 means that the

Server allows two echo messages to be missed. However, if the third is not received, the Server assumes the connection is broken.

The time period associated with BCTMO is equal to the BCTMO value times the PGRAN value (see **Setting Miscellaneous Parameters** below for the PGRAN description). In other words, if the Server does not receive the echo message within this time period, it assumes the connection is broken.

5.1.2 Defining Transaction Limits

Use the transaction-related parameters to adjust individual node characteristics and reconcile issues over

- network versus local memory usage (NSESS and NOIP)
- local versus remote load limitations (NLOGR and NLOGS)
- local versus network performance (NSSP and NASE)

The values you select for the seven transaction limits are node-type dependent; that is, the values for servers, requesters, and combination requester/servers are likely to be different.

NSESS

The number of connections (sessions) specifies the maximum number of virtual circuits you can have between this and other nodes. The parameter is valid for server and requester nodes and applies to connections established by the local Network Resource Manager to a remote Server Driver and between local and remote NET: devices. Each connection requires 256 bytes from the FlexNet memory pool (see NIMP below). Note that log ons by multiple processes on the same requester node to the same server node use the same connection.

Connections are allocated dynamically as requested, and there is no internal mechanism that delimits the connections available for NET: device versus requester/server communication. Consequently, network applications that establish multiple connections through the NET: device can use up the entire allocation and prevent requester/server communications until a connection is available. When the maximum has been reached, the next connection request receives the E_CONFAIL error code.

NOIP

The number of operations in progress sets the maximum number of on-going operations a requester can have with all servers. Each operation requires 128 bytes from the FlexNet memory pool (see NIMP below).

The total number of operations are shared among all process families. (Each physical and virtual console on a FlexOS node is identified by a unique process family ID.) Operations are allocated on a first come, first served basis. When the maximum has been reached, the next request receives the E_NO_OIP error code.

NLOGR

The number of log ons per requester defines how many servers can be logged on per process family. NLOGR differs from NOIP in that NOIP is applied on a per node rather than per process family basis. Specify a zero value to impose no limit. When you specify a limit, the E_LLIMIT error code is returned when the user attempts to exceed it.

Set NLOGR to prevent users from taking up all resources.

NLOGS

The number of log ons per server defines how many requesters can log on the node simultaneously. (NLOGS is meaningless on requester-only nodes.) Specify a zero value to impose no limit. When you specify a limit, remote requesters get the E_LOGREFUSED error code when it is exceeded.

Use NLOGS to regulate the load on the server. The issue to consider on server-only nodes is how many requesters can be serviced efficiently. An important factor in this consideration is the number of synchronous server processes (NSSP) and asynchronous events (NASE) set. For server nodes with few processes and events, it might be appropriate to minimize the number of log ons so that you do not end up with the bottleneck of many requesters queued and waiting for a limited resource.

NSSP

The number of synchronous processes defines how many processes are created on the node to service remote synchronous Supervisor calls. (NSSP is meaningless on requester-only nodes.) Note that calls received when no process is available are queued on a FIFO basis.

The issue to consider when selecting the NSSP is the tradeoff of network versus local performance. That is, the more processes there are available to service a call, the less likely it is that the requester must wait for a process to become available. However, the more processes there are to service the call, the longer it takes to execute each request.

NASE

The number of asynchronous events defines how many remote asynchronous Supervisor calls can be serviced simultaneously by the node. (NASE is meaningless on requester-only nodes.) The maximum value is 24. Requests received when no event is available are queued on a FIFO basis. The issue regarding the NASE value is the network versus local performance tradeoff described in NSSP.

5.1.3 Establishing Memory Requirements

You can adjust three areas of FlexNet's memory usage: the size and number of large message buffers (BSIZE and NBUF), the amount of memory used by the Name Server Driver to store the names database entries (NNTE), and the FlexNet internal memory pool (NIMP) used to store the transient data associated with connections and operations in progress.

BSIZE

FlexNet uses a large buffer for all messages larger than 256 bytes. If the message is larger than the buffer, FlexNet uses the same buffer. Multiple buffers are never assigned for the same message.

The buffer size you select affects message throughput and network response time. The larger the message buffer the more kilobytes per second are transferred. However, the larger the buffer, the higher the message response time.

Typically, the curve produced by plotting kilobytes per second for different message sizes climbs rapidly and levels off. The actual values are a function of the network packet size, transmission speed, and message overhead. Plotting response time against message size tends to produce a linear progression.

Other factors you should consider when selecting the large buffer size are average message size and message frequency. For example, you would want larger buffers for infrequent large file transfers while smaller buffers would be more appropriate for frequent file record updates.

NBUF

The number of large buffers the node requires depends on its message load. Note, however, that four is the minimum. If a buffer is not available, FlexNet waits for another operation to complete and release a buffer. FlexNet does not return an error when a large buffer is not available.

NNTE

The number of name table entries determines how many entries from the names database file are stored in memory. The amount of memory used is a multiple of the number specified times the size of each entry.

The Name Server Driver stores name in the buffer on a first-use basis as the translations are read from the disk-based database file by the NAMES utility. , the entries are recorded in the buffer.

NIMP

FlexNet requires a minimum 2K bytes to maintain static internal data structures. The remaining allocation is dynamically allocated to connections and operations in progress.

To determine a node's requirement, add the 2K minimum to the following:

For the requester side:

- 256 bytes for each connection (NSESS) allowed
- 128 bytes for each operation in progress (NOIP) allowed

For the server side:

- 256 bytes for each simultaneous request

The memory blocks are allocated on a first come, first served basis as processes make RECEIVE message calls. The space for operations in progress is acquired when the process makes a SEND call. Requester processes get the E_MEMORY error code ORed with the Network Resource Manager module number (8060H in the high order word) when no memory is available in the pool.

Regarding the server requirements: you do not need to have a block of memory allocated for each connection. However, under peak load this creates a situation in which a server process might not have access to its connection until a block becomes free. Until a block becomes free, the server process can neither accept nor reject incoming requests.

5.1.4 Setting Miscellaneous Parameters

Both of the miscellaneous parameters are security-related. Use the EXTSEC parameter to select extended or normal security on server nodes and the PKEY parameter to define the password encryption key.

EXTSEC

Only bit 0 of the EXTSEC byte is used; bits 1 through 7 are reserved and must be zero. Different server nodes can have different security levels. However, once selected, normal or extended security measures are applied to all requesters attempting to log on.

Set bit 0 to 1 to select extended security. When extended security is enabled, all users who want access to the node must make explicit LOGON commands specifying the node name, the account name, and the password. The server process runs LOGON to compare the request against the node's USER.TAB entries.

Set bit 0 to 0 to select normal security. When the server has normal security set, the server process checks only for an account entry in the USER.TAB file; it does not perform a password check.

The requester/server log on dialog consists of a series of messages that either automatically or explicitly qualify the user and establish a connection. See Appendix A for the description of the log on dialog and the message contents.

PKEY

The PKEY value is a 10-byte encryption key that modifies FlexNet's password encryption algorithm. If you do not specify an encryption key, FlexNet encrypts and decodes the password regardless. All nodes in communication with each other must have the same PKEY value. You can, however, have groups of nodes on the same network with different encryption keys.

The PKEY specification can take two forms:

- To specify a number, precede the value by 0x. For example, to use the hexadecimal value B8 as the encryption key, the PKEY entry would be 0xB8.
- To specify a string, enclose the entry in quotation marks. For example, to use the word MARY as the encryption key, your PKEY entry would be "MARY".

If you use NETSET to set PKEY, it pads your entry with spaces to complete the 10-byte requirement. If you use SET, be sure to provide all 10 bytes.

5.2 NETSET Description

Use the NETSET command to make changes to the NETPARAM table. All changes must be made before the local node name is set with the NAMES utility. Attempts to make changes after that return error. You can, however, call GET to get the NETPARAM table (NETPARAM table number is 60H) after LOCAL has been set. Typically, NETSET and NAMES should be run in the boot script.

You NETPARAM values by their name either from the command line or from a file. The NETSET command line is as follows:

```
NETSET parm=value parm=value ...
```

Use the parameter's NETPARAM mnemonic as the parm argument. Separate each equation with a space to set multiple parameters with the same command. For example, the following command sets the maximum number of connections, number of remote log ons, the size of the large buffers, and the number of name table entries.

```
NETSET NSESS=8 NLOGS=2 BSIZE=2 NNTE=10
```

The parameter order is not significant.

To set the network parameters from a file, enter the command

```
NETSET FILE filename
```

where filename is the resource file name. Use an editor to create the resource file. Be careful that the editor does not insert any special characters.

Set parameters within the file using the same mnemonics and syntax used in the NETSET command line. Separate each expression with a space or a carriage return.

There are two other NETSET options:

NETSET LIST	Displays the current NETPARAM table values.
NETSET HELP	Displays a summary definition of the NETPARAM fields.

Both commands list the parameters by the mnemonics you use in the NETSET resource file and command line.

End of Section 5

Message Contents

Messages between the FlexNet Network Resource Manager and Server Driver are constructed according to the rules specified in the IBM Server Message Block (SMB) data format protocol. The protocol dialect between FlexOS nodes is DRNET2.0. The protocol dialect between FlexOS and PC DOS nodes is PC NETWORK PROGRAM 1.0.

This appendix describes the components of the SMB protocol, the connection dialog used to reconcile SMB dialects and log on accounts, and the message contents under the DRNET2.0 dialect.

A.1. SMB Format Description

The Server Message Block message has three basic components:

- a fixed-length header
- a sequence of zero or more 16-bit parameters
- a variable length data field

Table A-1 lists the SMB fields used under the DRNET2.0 dialect. Note that the fields are implemented somewhat differently under this dialect than under the "PC NETWORK PROGRAM 1.0" dialect.

Table A-1. Server Message Block Format

Field Name	Length (in bytes)	Value
Header		
ID	1	0xFF
SERVER	3	ASCII characters SMB
SMBFUNC	1	Message function code (see message descriptions for the specific values)
RETCLASS	1	0x00 (this byte is used only by servers running the IBM PC LAN Program)
Reserved	1	
UID	2	Process family ID
FLAGS	1	Bit 7 indicates direction as follows: If 0, message is request If 1, message is response Flag bits 0 through 6 must be 0.
Reserved	14	

Table A-1. (Continued)

Field Name	Length (in bytes)	Value
PID	4	In request messages, ID of process initiating SVC. In response messages, SVC return code.
Reserved	2	
TID	2	Transaction ID
Parameters		
PARMCNT	1	Number of parameters in message
P1	2	First parameter
P2	2	Second parameter
.		
.		
Pn	2	Last parameter
Data		
BUFLEN	2	Total length of data buffer
DBUF		First byte of each block indicates the data type as follows: 01H: Variable length server data block 02H: Dialect ID string 03H: ASCIIZ path name string 04H: ASCIIZ string 05H: Function-specific data block For types 01H and 05H, indicate data length with an unsigned 16-bit integer immediately following type byte. Terminate data structures with a NULL.

WORD data items are sent low byte first; LONG data items are sent low word first and low byte first within each WORD.

Note: WORD and LONG values may be aligned on odd-byte boundaries.

The ID, SERVER, and RETCLASS fields have the same value in all messages. The message descriptions which follow indicate the values for the other fields.

The SMB DRNET2.0 functions provide the network and file system operations listed in Table A-2. The message descriptions follow the table. In the descriptions, long SVC parameters (for example, the CREATE SVC size parameter and the GET SVC ID parameter) are transferred as SMB parameters low order word first and the parenthetical value for each DBUF item indicates its type value.

Table A-2. Server Message Block DRNET2.0 Functions

Function Name	Number	Description
Network Functions:		
ECHO	75H	Determine if server is up
END OF PROCESS	11H	Close files of specified process
LOGOFF	74H	Log off remote account and end connection
LOGON	73H	Establish connection and log on remote account
VERIFY DIALECT	72H	Determine server SMB dialect
File System Functions:		
CANCEL	82H	Cancel network transaction
CLOSE	04H	Close a remote file
CREATE	03H	Create a remote file or directory
DELETE	06H	Delete a remote file
DEVLOCK	83H	Lock a remote device
GET	84H	Retrieve remote table information
LOCK	0CH	Lock or unlock a remote file
LOOKUP	86H	Retrieve remote table information
OPEN	02H	Open a remote file
READ	0AH	Get data from an open, remote file
RENAME	07H	Rename a remote file
SEEK	12H	Set or determine value of remote file pointer
SET	87H	Set remote table information
SPECIAL	88H	Perform SPECIAL function on a remote file
WRITE	0BH	Write data to an open, remote file

A.2. Network Function Messages

FlexNet functions establish a common dialect, log on and off remote accounts, and ensure the validity of an existing connection.

ECHO

Request message to server		Response message from server	
SMBFUNC:	75H	SMBFUNC:	75H
UID:	0	UID:	0
FLAGS:	00	FLAGS:	80H
PID:	0	PID:	0
TID:	Transaction ID	TID:	Same as request
PARMCNT:	0	PARMCNT:	0
BUFLEN:	0	BUFLEN:	0

END OF PROCESS

Request message to server		Response message from server	
SMBFUNC:	11H	SMBFUNC:	11H
UID:	0	UID:	0
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	0	PARMCNT:	0
BUFLEN:	0	BUFLEN:	0

LOGOFF

Request message to server		Response message from server	
SMBFUNC:	74H	SMBFUNC:	74H
UID:	Family ID	UID:	Family ID
FLAGS:	00	FLAGS:	80H
PID:	0	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	0	PARMCNT:	0
BUFLEN:	0	BUFLEN:	0

LOGON

Request message to server		Response message from server	
SMBFUNC:	73H	SMBFUNC:	73H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	0	PID:	Return Code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	4	PARMCNT:	5
P1:	Password included flag	P1:	Max. buffer size
P2:	Max. buffer size	P2:	Max. buffer size
P3:	Max. buffer size	P3:	User ID
P4:	Broken connection timeout	P4:	Group ID
BUFLEN:	Name + password	BUFLEN:	0
DBUF:	Account name (04) Encrypted password (01)		

In the log on request message, the requester indicates in parameters P1 through P4 whether a password is included, its maximum message buffer size (a LONG value, low byte first), and broken connection timeout (BCTMO value from the NETPARAM table). The password is included only when the user initiates the log on by running the LOGON program. If the log on attempt is initiated by a remote node reference in a path specification, no password is included in the message. The request message data buffer always contains the local account name

The server's log on response message indicates the maximum buffer size supported by the server (a LONG value, low byte first) and the account's user and group IDs on the server. These can be different than the account's corresponding IDs on the local node.

VERIFY DIALECT

Request message to server		Response message from server	
SMBFUNC:	72H	SMBFUNC:	72H
UID:	0	UID:	0
FLAGS:	00	FLAGS:	80H
PID:	0	PID:	0
TID:	0	TID:	0
PARMCNT:	0	PARMCNT:	1
		P1:	Zero-based index to selected dialect (see below) or FFFF if server supports none in the list
BUFLEN:	Length of dialect list	BUFLEN:	0
DBUF:	List of variable-length, null-terminated dialect names. Precede each name with type-byte value 02.		

The VERIFY DIALECT dialog determines if the requester and server support the same SMB dialects. The requester sends a list of the dialects in which it can communicate. The server picks the dialect it speaks from the list and sends back a value which indicates its selection. The value is derived by multiplying the dialect's offset (zero-based) from the first entry by two. For example, in a list containing three dialects, the offset and values would be as follows:

	Offset	Index Value
First dialect	0	0
Second dialect	1	2
Third dialect	2	4

A.3. File System Functions Messages

Generally, the request message parameters P1 through Pn contain the Supervisor call parameters. In addition, the message's parameter sequence corresponds to the SVC parameter sequence. For example, parameter P1 always contains the contents of the Supervisor call's mode byte (indicating whether the synchronous [0] or asynchronous

[1] form of the SVC has been called) in the high order byte and the option byte in the low order byte. Variable length data--for example, path specifications or table names--are transferred in the message's data buffer.

The response message uses the PID field to indicate the Supervisor call's return code. This word never contains data--possible values are zero or the negative error code. When the call returns a positive value (for example, READ returns the number of bytes read, CREATE and OPEN return a file number), the value is returned in the response message's parameters rather than the PID. Variable length data is returned in the response message's data buffer.

These rules apply to all the message descriptions below. There are, however, some cases where additional data is transferred or some fields are omitted. The message descriptions include explanations when the contents do not logically follow the SVC parameters.

CANCEL

Request message to server		Response message from server	
SMBFUNC:	82H	SMBFUNC:	82H
UID:	0	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	1	PARMCNT:	0
P1:	TID of operation to cancel		
BUFLEN:	0	BUFLEN:	0

CLOSE

Request message to server		Response message from server	
SMBFUNC:	04H	SMBFUNC:	04H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	4	PARMCNT:	0
P1:	Option byte value		
P2:	Flags value		
P3:	File number		
P4:	File number		
BUFLEN:	0	BUFLEN:	0

CREATE

Request message to server		Response message from server	
SMBFUNC:	03H	SMBFUNC:	03H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	6	PARMCNT:	4
P1:	Option byte value	P1:	Access flags
P2:	Flags	P2:	Table type
P3:	Record size	P3:	File number
P4:	Security word	P4:	File number
P5:	File size	P5:	Record size
P6:	File size		
BUFLEN:	Length of path	BUFLEN:	0
DBUF:	Path specification (03)		

The request message parameter values come from the Supervisor call. Besides the new file's file number, the response message also includes the file's access flags indicating the owner, group, and world access privileges and the file's table type (for example, a pipe file would have table number 10H, a disk file number 20H).

DELETE

Request message to server		Response message from server	
SMBFUNC:	06H	SMBFUNC:	06H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	2	PARMCNT:	0
P1:	Option byte value		
P2:	Flags value		
BUFLEN:	Length of path	BUFLEN:	0
DBUF:	Path specification (03)		

DEVLOCK

Request message to server		Response message from server	
SMBFUNC:	83H	SMBFUNC:	83H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	4	PARMCNT:	0
P1:	0		
P2:	Option word value		
P3:	File number		
P4:	File number		
BUFLEN:	0	BUFLEN:	0

GET

Request message to server		Response message from server	
SMBFUNC:	84H	SMBFUNC:	84H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	6	PARMCNT:	0
P1:	Table number		
P2:	Flags value		
P3:	id		
P4:	id		
P5:	bufsize		
P6:	bufsize		
BUFLEN:	0	BUFLEN:	bufsize
		DBUF:	Table information (01)

LOCK

Request message to server		Response message from server	
SMBFUNC:	0CH	SMBFUNC:	0CH
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	8	PARMCNT:	0
P1:	Mode and option bytes		
P2:	Flags value		
P3:	File number		
P4:	File number		
P5:	offset		
P6:	offset		
P7:	nbytes		
P8:	nbytes		
BUFLEN:	0	BUFLEN:	0

LOOKUP

Request message to server		Response message from server	
SMBFUNC:	86H	SMBFUNC:	86H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	8	PARMCNT:	2
P1:	Table number	P1:	Number of items found
P2:	Flags value	P2:	Number of items found
P3:	bufsize		
P4:	bufsize		
P5:	itemsize		
P6:	itemsize		
P7:	key		
P8:	key		
BUFLEN:	Length of name	BUFLEN:	bufsize
DBUF:	name (04)	DBUF:	Table data of matching items (01)

OPEN

Request message to server		Response message from server	
SMBFUNC:	02H	SMBFUNC:	02H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	2	PARMCNT:	4
P1:	Option byte	P1:	Access flags
P2:	Flags value	P2:	Table type
		P3:	File number
		P4:	File number
		P5:	Record size

The access flags returned indicate actual access granted to allow use of reduced access flag on remote opens.

READ

Request message to server		Response message from server	
SMBFUNC:	0AH	SMBFUNC:	0AH
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	9	PARMCNT:	1
P1:	Mode and option bytes	P1:	Number of bytes read
P2:	Flags		
P3:	File number		
P4:	File number		
P5:	Byte count		
P6:	offset		
P7:	offset		
P8:	Bytes remaining		
P9:	Number of delimiters		
BUFLEN:	Length of delimiter array	BUFLEN:	Length of file data
DBUF:	Delimiter array (01)	DBUF:	File data (01)

The READ SVC's bufsize specification is not passed to the server because the maximum message length is 64K bytes. Instead, the size specification is expressed in parameters P5 and P8 as bytes to be read and bytes remaining to be read, respectively. The Network Resource Manager handles the addition and subtraction and ensures that the SVC returns to the calling process with the assembled data rather than the individual blocks.

RENAME

Request message to server		Response message from server	
SMBFUNC:	07H	SMBFUNC:	07H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	2	PARMCNT:	0
P1:	0		
P2:	Flags		
BUFLEN:	Length of old and new path specifications	BUFLEN:	0
DBUF:	current name (03) new name (03)		

SEEK

Request message to server		Response message from server	
SMBFUNC:	12H	SMBFUNC:	12H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	6	PARMCNT:	2
P1:	0	P1:	Current location
P2:	Flags	P2:	Current location
P3:	File number		
P4:	File number		
P5:	offset		
P6:	offset		
BUFLEN:	0	BUFLEN:	0

SET

Request message to server		Response message from server	
SMBFUNC:	87H	SMBFUNC:	87H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	6	PARMCNT:	0
P1:	Table number		
P2:	Flags		
P3:	id		
P4:	id		
P5:	bufsize		
P6:	bufsize		
BUFLEN:	bufsize	BUFLEN:	0
DBUF:	Table information (01)		

SPECIAL

Request message to server		Response message from server	
SMBFUNC:	88H	SMBFUNC:	88H
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	11	PARMCNT:	3
P1:	Mode and function number	P1:	More flags (see below)
P2:	Flags	P2:	dbufsize for this message
P3:	File number	P3:	pbufsize for this message
P4:	File number		
P5:	More flags (see below)		
P6:	dbufsize		
P7:	dbufsize		
P8:	pbufsize		
P9:	pbufsize		
P10:	dbufsize for this message		
P11:	pbufsize for this message		
BUFLEN:	Sum of P10 and P11	BUFLEN:	Sum of P2 and P3

DBUF:	databuf data (if any) (01) parmbuf data (if any) (01)	DBUF:	databuf data (01) parmbuf data (01)

The **More flags** word in the request message is a bit map of flags indicating the following:

Bit 0: 0 = Not first message
1 = First message in a series of SPECIAL calls

Bit 1: 0 = Not last message
1 = Last message in a series of SPECIAL calls

Bit 2: 0 = dbufsize value (P6 and P7) is size of data
1 = dbufsize value is data

Bit 3: 0 = pbufsize value (P8 and P9) is size of data
1 = pbufsize value is data

Bits 4 through 15 are reserved.

The **More flags** word in the response message makes use of bits 0 and 1 only.

The Network Resource Manager and Server Driver manage the conversion of single SPECIAL call to multiple SPECIAL messages when the data or parameters exceed 64K bytes in size.

WRITE

Request message to server		Response message from server	
SMBFUNC:	0BH	SMBFUNC:	0BH
UID:	Family ID	UID:	--
FLAGS:	00	FLAGS:	80H
PID:	Process ID	PID:	Return code
TID:	Transaction ID	TID:	Same as request
PARMCNT:	8	PARMCNT:	2
P1:	Mode and option bytes	P1:	Bytes written
P2:	Flags	P2:	Bytes written
P3:	File number		
P4:	File number		
P5:	Byte count		
P6:	offset		
P7:	offset		
P8:	Bytes remaining		
BUFLEN:	Length of data to write	BUFLEN:	0
DBUF:	Data (01)		

See the READ message description for the explanation of the Byte count and Bytes remaining parameters

End of Appendix A

PC DOS Node Support

FlexNet includes two loadable drivers providing communication services to nodes running the IBM PC LAN Program under PC DOS 3.1. The FlexNet DOS Server Driver translates messages from PC DOS requester nodes for processing under FlexOS and the FlexNet DOS Requester Driver prepares messages for processing on PC DOS server nodes. Both the DOS Server and Requester drivers support the "PC NETWORK PROGRAM 1.0" dialect.

This appendix describes the PC DOS and FlexOS functions supported on a remote basis by these FlexNet drivers. The explanation of each driver describes how mismatches between FlexOS and PC DOS calls are resolved and what limitations are enforced.

Note: For FlexOS and PC DOS Network Program nodes to be present on the same network, the DRISRVR socket must have address 20H and DRIRQSTR socket must have address 00H.

B.1. FlexNet PC DOS Server Driver

The FlexNet PC DOS Server Driver supports connection and disk file system transactions from nodes running the PC DOS operating system. Not supported are the message communication and remote program loading (booting) functions. Table B-1 lists the SMB functions supported by the PC DOS Server Driver. Errors that occur during execution of the function are translated from their FlexOS value to an equivalent PC DOS code.

Table B-1. SMB PC NETWORK PROGRAM 1.0 Functions Supported

Value	Command	Value	Command
00H	Create Directory	0DH	Unlock Byte Block
01H	Delete Directory	0EH	Create Temporary File
02H	Open File	0FH	Create New File
03H	Create File	10H	Check Directory
04H	Close File	11H	End of Process
05H	Commit All Files	70H	Start Connection
06H	Delete File	71H	End Connection
07H	Rename File	72H	Verify Dialect
08H	Get File Attribute	80H	Get Disk Attribute
09H	Set File Attribute	81H	Search Multiple Files
0AH	Read Byte Block	C0H	Create Spool File
0BH	Write Byte Block	C1H	Spool Byte Block
0CH	Lock Byte Block	C2H	Close Spool File

The FlexOS OPEN flags do not correspond directly to the PC DOS open modes. Table B-2 lists the four PC DOS mode fields and indicates how they are implemented on the FlexOS Server.

Table B-2. PC DOS to FlexOS OPEN Modes

PC DOS Open Mode Name	Value	FlexOS OPEN SVC Flag Bit Value
Inheritance Flag		Ignored
Sharing Mode	000: Compatibility	Bit 4 = 1 (Shared)
	001: Exclusive	Bit 4 = 0 (Exclusive)
	010: Deny write	Bit 4 = 1 (Shared) Bit 5 = 1 (Shared reads only)
	011: Deny read	Bit 4 = 0 (Exclusive)
	100: Deny none	Bit 4 = 1 (Shared)
Access Privilege	000: Read	Bit 3 = 1 (Read access, Bits 0 - 2 are 0)
	001: Write	Bit 2 = 1 (Write access, Bits 0, 1, and 3 are 0)
	010: Read/write	Bits 2 and 3 are 1 (Read/write access bits 0 and 1 are 0)

Note also that the RENAME functions are performed differently when the command line includes wildcards sometimes producing unpredictable results. Consequently, we recommend against the use of wildcards to rename files from PC DOS requesters on FlexOS servers and vice versa.

PC DOS requesters must have an account in the FlexOS server's USER.TAB to get access to the node. In his or her NET USE command, the user must specify the account with a logical name (also referred to as a shortname); directories and paths are not valid. Once the log on is established, FlexNet and FlexOS provide the same level of transparent access provided by the IBM PC LAN Program.

The USER.TAB entry for a PC DOS requester has the same format as the FlexOS user's. However, some of the fields are interpreted differently for the PC DOS requester than they are for the FlexOS user. Use the following guidelines when making each entry.

username	Use the logical network path name (shortname) that is sent in the SMB Start Connection message
password	Use the password that is sent in the SMB Start Connection message.
userid	Give the PC DOS requester a userid consistent with your FlexOS numbering scheme. The value has meaning on the FlexOS node only; it is not returned to the PC DOS node.
groupid	Give the PC DOS requester a groupid consistent with your FlexOS numbering scheme. The value has meaning on the FlexOS node only; it is not returned to the PC DOS node.
home	Specify the user's root directory path. User access to the disk is limited to this directory and its subdirectories. Prefix substitution is not allowed. You must use the name given to the device in the DVRLOAD or DVRLINK command used to install it.
wmanager	Leave this field empty if the user is to have remote access only. Fill in the field if the user is to have local and remote access.
shell	Leave this field empty if the user is to have remote access only. Fill in the field if the user is to have local and remote access.
access	Set the value to 2 for remote-only access or 3 for local and remote access.

The PC DOS Server Driver appends the device name and path in the account's home field to the beginning of all path specifications from the user. In addition, the PC DOS Server Driver does not allow prefix substitution. Thus, the user has access to the directory specified in the home entry and its subdirectories only.

B.2. FlexNet PC DOS Requester Driver

The PC DOS Requester Driver supports the remote execution of FlexOS disk-file-related Supervisor calls on server nodes running PC DOS. The messages to the server are formatted according to the PC NETWORK PROGRAM 1.0 protocol. Table B-3 lists the compatible calls. See the FlexNet User's Guide for the list of FlexOS commands supported over the network.

Note: PC DOS and FlexOS treat exclusive opens differently. FlexOS allows multiple OPEN calls by the process that opened it initially. PC DOS allows no further OPEN calls until the file is closed. Consequently, close files on PC DOS nodes opened in exclusive mode before attempting another OPEN call. Similarly, you must close a file you created before you can open it.

Table B-3. Supervisor Calls Executable on PC DOS Servers

SVC	Support	
CLOSE		
Flags:	Flush only	Yes
	Fully close	Yes
CREATE		
Options:	Disk file	Yes (but no pipes)
	Disk directory	Yes
	Virtual console	No
Flags:	Delete/Set access	Yes
	Write access	Yes
	Read access	Yes
	Shared access	Yes (Exclusive = Deny R/W)
	Shared reads	Yes (Shared R/W = Deny none; Shared R/O = Deny write)
	Shared file pointer	Yes
	Temporary	Yes
	Contiguous allocation	No
	Delete if existing	Yes
	Use specified security	No
	Force case	Yes (always uppercase)
	Literal name	Yes
Arguments:	Record size	0 or 1 only
	Security word	Ignored
	Size	0 only
DELETE		
Flags:	Force case	Yes (always uppercase)
	Literal name	Yes

Table B-3. (Cont'd)

SVC	Support	
GET		
Option:	Table	DISKFILE table only
Flags:		Must be 0
Arguments:	id	Ignored
	Buffer	Yes (only partial info; see DISKFILE Table Accommodations)
LOCK		
Flags:	Unlock	Yes
	Exclusive lock	Yes
	Exclusive write lock	No
	Shared write lock	No
	Return error on conflict	Yes
	Wait on conflict	Yes
	Offset relative to start of file	Yes
	Offset relative to EOF	Yes
	Offset relative to file pointer	Yes
Arguments:	swi	Yes
	fnum	Yes
	offset	Yes
	nbytes	Yes, but you cannot overlap locked areas and you must match nbyte value in related lock and unlock commands

Table B-3. (Cont'd)

SVC	Support	
LOOKUP		
Options:	Table	DISKFILE table only
Flags:	Include hidden files	Yes
	Exclude hidden files	Yes
	Include system files	Yes
	Exclude system files	Yes
	Include directory files	Yes
	Exclude directory files	Yes
	Include volume label	Yes
	Exclude volume label	Yes
	Include normal files	Yes
	Exclude normal files	Yes
	Force case	Yes (always uppercase)
	Literal name	Yes
Arguments:	Name	Yes
	Buffer	Yes (only partial info; see DISKFILE Table Accommodations)
	itemsize	Yes
	nfound	Yes
	key	Yes
OPEN		
Flags:	Delete/Set	Yes
	Execute	Yes
	Read	Yes
	Write	Yes
	Share	Yes
	Exclusive	Yes (Deny R/W)
	Share R/W	Yes (Deny none)
	Share read	Yes (Deny write)
	Share file pointer	Yes
	Reduced access	Yes
	Force case	Yes (always uppercase)
	Literal name	Yes

Table B-3. (Cont'd)

SVC	Support	
READ		
Flags:	Flush device buffer	Yes
	Read until delimiter	Yes
	Non-destructive read	Ignored
	Pre-initialized read	No
	Delimiter included	Yes
	Edited read	No
	Offset relative to start of file	Yes
	Offset relative to EOF	Yes
	Offset relative to file pointer	Yes
Arguments:	swi	Yes
	fnum	Yes
	bufsize	Yes
	offset	Yes
	delimiters	Yes
RENAME		
Flags:	Force case	Yes
	Literal name	Yes
Arguments:	name	Yes
	newname	Yes
SEEK		
Flags:	Offset relative to start of file	Yes
	Offset relative to EOF	Yes
	Offset relative to file pointer	Yes
Arguments:	offset	Yes

Table B-3. (Cont'd)

SVC	Support	
SET		
Options:	Table	DISKFILE table only
Flags:	Must be 0	
Arguments:	id	Ignored
	Buffer	Yes (only partial info.; see DISKFILE Table Accommodations below)
WRITE		
Flags:	Flush buffer	Yes
	Truncate	Yes
	Offset relative to start of file	Yes
	Offset relative to EOF	Yes
	Offset relative to file pointer	Yes
Arguments:	swi	Yes
	fnum	Yes
	bufsiz	Yes
	offset	Yes

The DOS Requester Driver provides both transparent log on and explicit log on to PC DOS servers. In both cases, the FlexOS account name must have a matching logical path name (also referred to as a shortname) on the server. Transparent log on is only provided for a logical path name matching the user's FlexOS username. Logical path names that do not match must be logged on explicitly.

Use the PC DOS NET SHARE command to establish the logical path names and passwords on the server. Once the log on is established, the DOS Requester Driver provides the same level of transparent access as the IBM PC LAN Program.

DISKFILE Table Accommodations

Although you can call GET, SET, and LOOKUP for the DISKFILE table from the FlexOS node, the "PC NETWORK PROGRAM 1.0" dialect's Get File Attribute (08H) and Set File Attribute (09H) commands do not support all the data fields in this table. Consequently, GET and LOOKUP calls do not provide all the table's data and not all of the table's parameters can be set with SET. Table B-4 lists the DISKFILE parameters supported.

Table B-4. DISKFILE Parameters Supported

Parameter	Description
KEY	Supported
NAME	Supported
ATTRIBUTE	All attributes except Security Enabled are supported for both GET and SET.
RECSIZE	Always 1 from GET and LOOKUP
USER	Not supported
GROUP	Not supported
PROTECT	Not supported
SIZE	Supported for GET
Time	Supported only for GET; the DOS Requester Driver converts the time format from PC DOS to FlexOS. PC DOS does not allow you to set the time.

End of Appendix B

Network Return Codes

Return codes generated by the Network Resource Manager and FlexNet drivers adhere to the format described in the FlexOS DOS Programmer's Guide. The source indicator (low order byte of the high word) values are assigned to the different network drivers as described in Table C-1. The numeric range for all network error codes (low order word) is 0x4380H to 0x43FFH. Table C-1 also lists the ranges assigned to each driver. Table C-2 lists the error codes common to all drivers. The driver-specific error codes follow this table listed by driver.

Table C-1. Network Error Source Codes

Code (in hex)	Driver	Range (in hex)
8060	Network Resource Manager	0x43A0 - 0x43AF
8061	Protocol Driver	0x4400 - 0x441F
8062	Transport Driver	0x4440 - 0x445F
8063	Server Driver	0x43B0 - 0x43DF
8064	NET: Device Driver	0x43E0 - 0x43FF
8065	Name Server Driver	0x4420 - 0x443F

Table C-2. Return Codes Common to All Drivers

Value	Mnemonic	Description
0L	E_SUCCESS	Operation was successful
0x4380L	E_BADFORM	Invalid XRB format
0x4381L	E_BADREQ	Bad request
0x4382L	E_NORESOURCE	No driver or memory available
0x4383L	E_CNCLD	Request was cancelled
0x4384L	E_BADNODE	No such node or buffer address

Table C-3. Transport Driver Error Codes

Value	Mnemonic	Description
0x4440L	E_REQTMO	Request timed out
0x4441L	E_MOREDATA	Message has more data
0x4442L	E_BADHANDLE	Invalid connection handle
0x4443L	E_BADCONN	Connection closed on pending request
0x4444L	E_REQDONE	Request already completed
0x4446L	E_NETERR	Network failure
0x4447L	E_BADSOCKET	Socket address does not exist
0x4448L	E_INUSE	Another node is using that name

Table C-4. Network Resource Manager Error Codes

Value	Mnemonic	Description
0x43A0L	E_NONODE	No such node
0x43A1L	E_CONFAIL	Cannot connect to node
0x43A2L	E_LOGFAIL	Unable to log-on remote node
0x43A3L	E_DIALECT	Remote node does not support any dialect supported by local node
0x43A4L	E_LOGGEDON	Already logged on to the node
0x43A5L	E_NOTLOGGED	Not logged on to node
0x43A6L	E_LOCNAME	Local node name not set
0x43A7L	E_LOGREFUSED	Remote log-on refused
0x43A8L	E_LOGPERM	Not permitted to log-on node
0x43A9L	E_LNAMESET	Local name already set
0x43AAL	E_LLIMIT	Process log-on limit exceeded
0x43ABL	E_NO_OIP	No operations in progress available to service this request

Table C-5. Name Server Driver Error Codes

Value	Mnemonic	Description
0x4420L	NS_ASYNC	Return successful--synch pending
0x4421L	E_NONAME	No such name was found
0x4422L	E_BADKEY	Key is unknown
0x4423L	E_DUPNAME	Name already exists
0x4424L	E_BADNAME	Invalid name specification
0x4425L	E_NOXPORT	Operation requires transporter and Transport Driver is not installed

Table C-6. NET: Device Error Codes

Value	Mnemonic	Description
0x43E0L	E_NOREMOTE	No remote node or socket specified
0x43E1L	E_RSVDSKT	Attempt to use reserved socket
0x43E2L	E_INVHANDLE	Invalid connection handle
0x43E3L	E_BADSTATE	Connection in wrong state
0x43E4L	E_TABLEFULL	Connection table is full

The Server and Protocol Drivers have the following error codes:

Server Driver:

0x43B0L	E_BROKEN	Internal error in remote module

Protocol Driver:

0x4400L	E_DOS	General error from PC DOS node

End of Appendix C

Index

O

W

X