H291 Bulk File Transfer (BFX™) Utility

for Bull GCOS8 Systems

Release 3.0.2

Software Reference Manual

MAN-REF-H291-3.0.2

Revision Record

Revision	Description
A (10/84)	Manual released at BFX version 2.0.
01 (02/87)	• Manual updated at release 2.1. • Installation section revised. • Publication number has been changed from 4399H291 to 460304.
02 (02/88)	• Revision packet 460497-01 to correspond to H291 release level 2.2. • The JID command and parameter were corrected. • Other minor editorial and technical changes were made.
03 (04/88)	• Manual updated at release 2.2. • This release added the FIXED, LNAME, and NSER commands; provided additional information on BLOCK, FILE, and MODE commands; and made changes to some of the command defaults. • New examples were added to the Application section; changes were made to the Bull Implementation Notes, and Abort Processing was added to the Appendix. • Minor technical and editorial changes were also included.
04 (07/92)	Manual updated to correspond to release 2.3. Removed all occurrences of GCOS III from the manual. Added the "Applications of BFX" and "H291 BFX Implementation Notes" sections. Updated the "Installation" section.
05 (01/2011)	Manual updated to correspond to release 3.0.
3.0.2 (9/2013)	Manual updated to correspond to release 3.0.2

You may submit written comments using the comment sheet at the back of this manual to:

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

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

support@netex.com

or, by visiting our web site at:

http://www.netex.com

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

Preface

This manual describes the user interface to Network Executive Software's ("NetEx Software") H291 Bulk File Transfer (BFX™) utility for GCOS8 operating systems. BFX is used in conjunction with NetEx Software's NETwork EXecutive (NETEX®) family of software products and NetEx Software's network hardware.

"Introduction" on page 1 describes BFX' including a network configurations which can support BFX, and the three programs which compose BFX and how they interact.

"Operator Interface" on page 11 presents user control statements and parameters. This section also shows the commands used when running BFX.

"Applications" on page 23 provides detailed examples of using BFX.

"H291 BFX Implementation Notes" on page 27 describes functionality and limitations distinctive to H291 GCOS8 BFX.

"Installation" on page 33 describes the installation of BFX.

"Appendix A. BFX Error Messages" on page 37 lists the BFX error messages.

"Appendix B. Abort Processing" on page 53 describes how the BFX user can control the actions of the process if a fatal error occurs during the processing of file transfers.

BFX uses NETEX, but the reader does not need to understand NETEX to use the first four sections of this manual and BFX.

The reader is assumed to be familiar with programming, using the GCOS8 operating system and the GCOS8 Command Language.

Reference Material

The following manuals contain related information.

Number	Title and Description
man-cnet-conf-mgr	*"C" Configuration Manager and NetEx Alternate Path Retry (APR) User Guide*
460280	*NCT Loader Software Reference Manual*
460580	*NETEX Application Programmer's Interface Software Reference Manual*

Notice to the Reader

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

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

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

Corporation Trademarks and Products

Network Executive Software **NetEx, BFX, PFX, USER-Access, TNP210**

These references are made for informational purposes only.

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

Notice to the Customer

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

Document Conventions

The following notational conventions are used in this document.

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

Contents

Figures

Tables

Introduction

The Network Executive Software ("NetEx Software") Bulk File Transfer (BFX™) utility is a software package to be used with NetEx Software's NETwork EXecutive (NETEX) software. BFX and NETEX provide the software to rapidly move large amounts of file data between computers. The computers may use different operating systems, provided they have the proper NETEX products installed (H291 BFX requires the H297 DX NETEX or H297IP NetEx/IP Requester product).

BFX can access any disk file format (although some may be accessible only in "BIT mode"), and most tape file formats supported by GCOS8. BFX is run as a batch job.

BFX can be used to transfer files between different hosts which may require specialized record or data conversion. BFX allows the use of NETEX or hardware assembly/disassembly and code conversion. For other code conversion, BFX has a set of user exits that allow user modules to process data both before and after transfer over the network. These user modules may take responsibility for accessing data. This makes it possible to copy and convert complex data between different machines, or copy direct access files or data bases.

Supported Configurations

BFX uses the NETEX communications subsystem and other network hardware for its data communications. It uses all of the capabilities of these products to move files at multi-megabit speeds over the following types of configurations:

- Local area networks.
- Multiple local area networks connected via wide area networks.

Sample Network Configuration

BFX requires that copies of the BFX programs and user-written job control language (JCL) that invoke BFX reside in both the source and destination CPUs, as shown in Figure 1. BFX may also be used in a way that allows the user to place all of the commands in one of the hosts (BFX must reside in both).

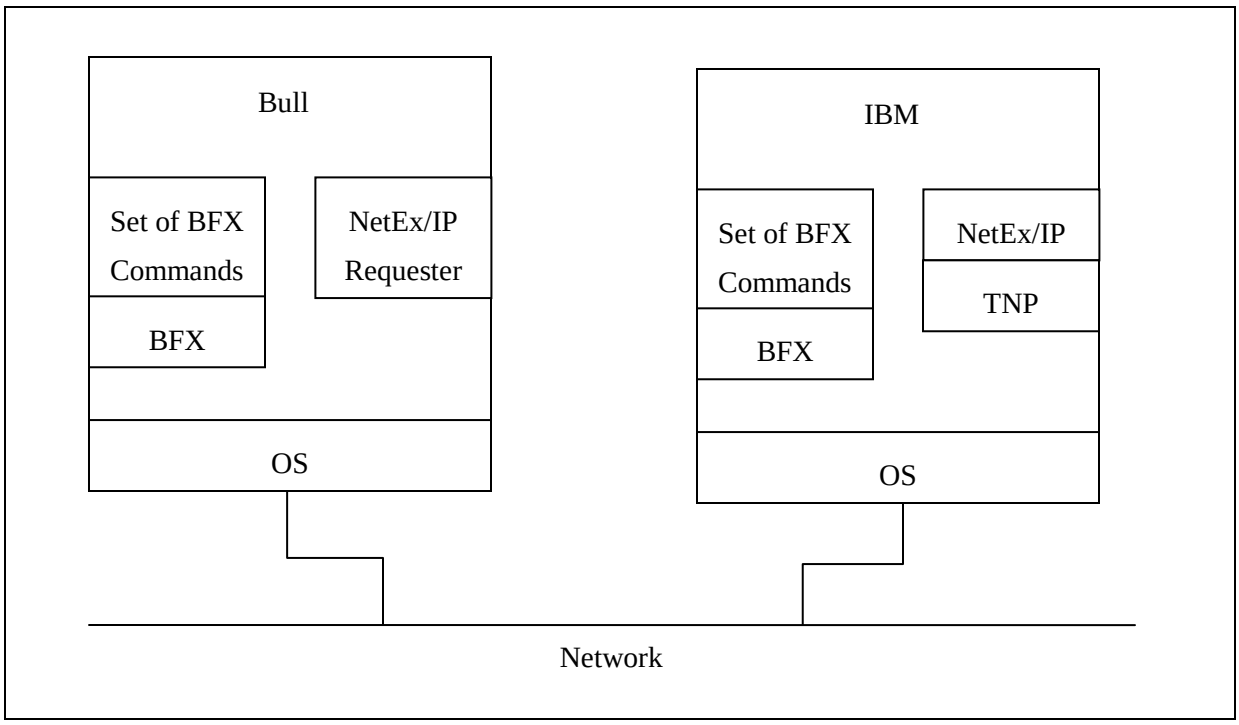

Figure 1. Sample BFX configuration.

Figure 1 is an example showing the following points:

- BFX and the NETEX or NETEX Requester (i.e. H297IP) program must reside in each host. If the two hosts use different operating systems, data transfer is still supported provided the proper versions of BFX and NETEX are installed and active. BFX uses NETEX and software code conversion, and allows user modules to perform more elaborate data conversion. NETEX Requesters must utilize a NETEX with the TNP feature (i.e. H210IPz and TNP210).
- Since the user directly invokes the BFX utility (which in turn calls NETEX), the BFX user does not need to learn to use NETEX itself.
- BFX is an application program that can be used like any other file processing program. It can be directly invoked by normal users without regard to the other BFX and NETEX applications that may be using NETEX at the same time.

Three BFX Programs

BFX is an application that consists of three separate components: BFX Transfer Initiator (BFXTI), BFX Transfer Responder (BFXTR), and BFX Job Submitter (BFXJS):

- The BFXTI program is used to SEND job files to BFXJS and to INITIATE file transfers from or to BFXTR.
- The BFXTR program RESPONDS to the BFXTI INITIATE to send or receive data files.
- The BFXJS program normally stays resident in the system. It INITIATES requests for job files to be submitted via BFXTI and spawns these jobs after they are successfully received. The job may be a BFXTR JCL file that will RESPOND to the BFXTI that sent it, or it may be a totally unrelated process such as a FILSYS activity, language processor, or any other job recognized by GCOS8.

All three of these programs can run in each host, making each host capable of initiating or responding to BFX requests.

Using BFX

To use BFX, the programmer must write two sets of commands and parameters. These sets will be called the local command set and the remote command set. The structure and contents of the local and remote sets vary according to the application. There are three basic applications

- Manual job submission
- Automatic job submission
- Remote job submission

The following pages describe the contents of these user procedures for each application, and how BFX processes the user requests.

Manual Job Submission

Manual job submission is the most basic application of BFX. When using manual job submission, the programmer must write a local JCL that uses BFXTI, and a remote JCL that uses BFXTR. Both the local JCL and the remote JCL are written using the control language and BFX commands for the host they will be executed on.

Both the local JCL and the remote JCL contain matched sets of SEND and RECEIVE BFX commands. A SEND in one JCL must match a RECEIVE in the other JCL, both jobs have the same identification and the responder specifies the host name of the initiator. The SEND command specifies the file to be read from the sending host. The RECEIVE command specifies the file to be written on the receiving host. The SEND and RECEIVE commands may also specify other information about the data being transferred.

To begin the file transfer, an operator on one host submits the local JCL and an operator on the other host submits the remote JCL. When the jobs are executed, the BFXTR program connects to the BFXTI program (via NETEX and the network) and the files are transferred.

Manual Job Submission Example

The following is a manual job submission example that is generalized so that the user can understand the general concept of this application. (An actual JCL includes other control language statements that are excluded from this example.) The NOSUBMIT command for BFXTI does not allow automatic job submission.

Assume that a user on the local host HOSTA wants to send file A to the remote host, and receive file B from the remote host using manual Job submission. The user writes the following local JCL:

```
$      IDENT   MYID,BFXTIJOB
$      USERID  MYID$PASS
$      SELECT  BFX/BFXTI
SEND TO HOSTB ID=BFXJOB FILE=FC*AA NOSUBMIT
RECEIVE FROM HOSTB FILE = FC*BB
$      PRMFL   AA,R,S,MYID/FILEA
$      PRMFL   BB,W,S,MYID/FILEB
$      ENDJOB
```

The “ID = BFXJOB” parameter uniquely identifies this connection and will also be used in the remote JCL. The user also writes the following remote JCL:

```
$      IDENT   HISID,BFXTRJOB
$      USERID  HISID$PASS
$      SELECT  BFX/BFXTR
RECEIVE FROM HOSTA ID=BFXJOB FILE=FC*A1
SEND TO HOSTA FILE=FC*A2
$      PRMFL   A1,W,S,HISID/FILEONE
$      PRMFL   A2,R,S,HISID/FILETWO
$      ENDJOB
```

Notice that a SEND in one JCL matches a RECEIVE in the other JCL.

Figure 2 illustrates the manual job submission process, using these sample local and remote files.

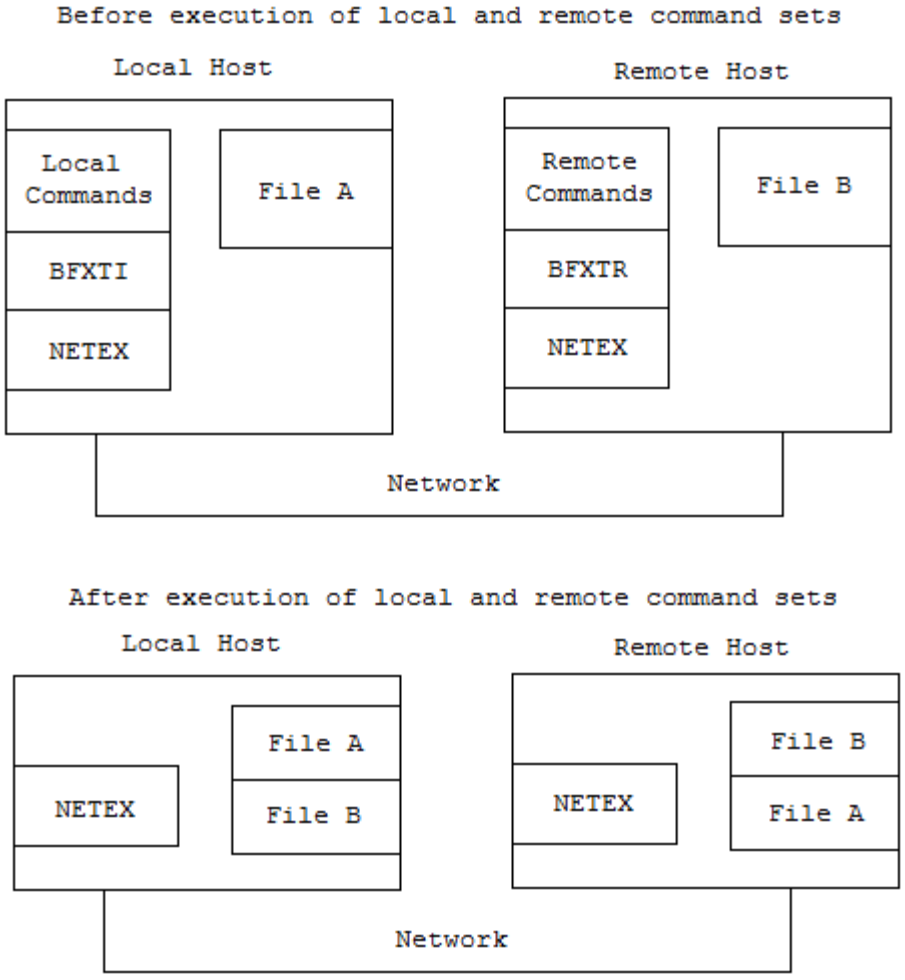

Figure 2. Manual Job Submission Example

Notice in Figure 2 that one JCL (the local JCL) uses BFXTI and the other JCL (the remote JCL) uses BFXTR. Execution of the local JCL and the remote JCL causes BFXTI and BFXTR to exchange the specified files using the network and NETEX.

Automatic Job Submission

Automatic job submission is the most common application of BFX. When using automatic job submission, the programmer must write the local JCL and the remote JCL which use commands suitable for each host. The JCL is similar to those used for manual job submission, using SEND and RECEIVE commands.

Instead of submitting· the local JCL and the remote JCL manually (as in manual job submission), the remote JCL is encapsulated .in the local JCL. The local JCL is then submitted for processing by BFXTI on the local host. BFXTI sends the remote JCL over the network to the BFXJS program on the remote host. BFXJS then automatically spawns this remote JCL for execution on the remote host. The local JCL and the remote JCL then transfer the specified files as they did for the manual job submission process.

Notice that no operator intervention is required on the remote host.

Automatic Job Submission Example

The following is an automatic job submission example that is generalized so that the user can understand the general concept of this application. (An actual JCL includes control language statements that are excluded from this example.)

Assume that a user on the local host wants to send file A to the remote host, and receive file B from the remote host using automatic job submission. The user writes the following JCL:

```
$      IDENT   MYID,BFXAUTO
$      USERID  MYID$PASS
$      SELECT  BFX/BFXTI
SEND TO HOSTB ID=BFXJOB MODE=ASCII FORMAT=GFRC FILE=FC*AA
RECEIVE FROM HOSTB FILE=FC*BB
$      PRMFL   JF,R,S,MYID/BFXTRJSL
$      PRMFL   AA,R,S,MYID/FILEA
$      PRMFL   BB,W,S,MYID/FILEB
$      ENDJOB
```

Note the contents of PRMFL BFXTRJSL (media code 2 card):

```
$      DUMMY   GENEW
$      IDENT   HISID,BFXTR
$      USERID  HISID$PASS
$      SELECT  BFX/BFXTR
RECEIVE FROM HOSTA ID=BFXJOB MODE=ASCII FORMAT=GFRC FILE=FC*Al
SEND TO HOSTA FILE=FC*A2
$      PRMFL   A1,W,S,HISID/FILEONE
$      PRMFL   A2,R,S,HISID/FILETWO
```

Notice remote JCL, the same remote JCL that was used in the previous example, is encapsulated in the local job command statements. This information in parentheses would be replaced by control language statements. See “Applications” on page 27 for more information.

Figure 3 on page 7 illustrates the automatic job submission process using these sample local and remote files.

Figure 3 shows that BFXTI first scans to see if there is a remote job to be submitted. When the remote job is found, BFXTI establishes a NETEX connection with BFXJS on the remote host. BFXTI then transfers the remote job to the BFXJS on the remote host.

After execution of the first part of the local JCL, BFXJS then spawns the JCL for execution on the remote host starting BFXTR. BFXTR then establishes a NETEX connection with the local host, and the specified files are transferred. While this job is running, BFXJS is ready to accept another job.

After execution of the local JCL and the remote JCL, the specified files have been transferred and are ready for use. The BFXTI, BFXTR, and BFXJS programs are ready for the next job.

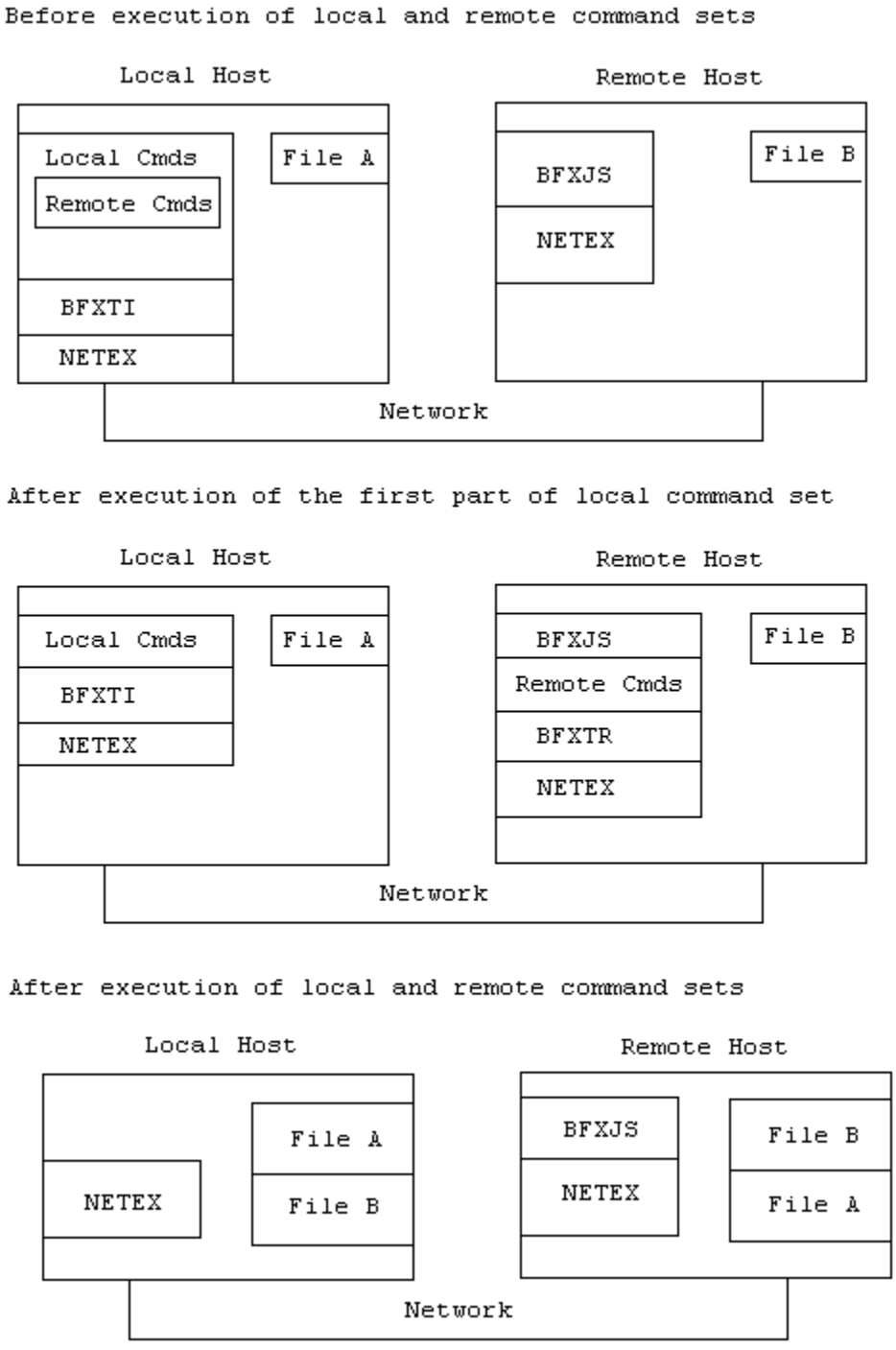

Figure 3. Automatic Job Submission Example

Remote Job Submission

Remote job submission is a special kind of automatic job submission. Using remote job submission, a user can submit a batch job to the remote host (instead of a JCL for BFXTR.) This batch job is then processed on the remote host.

Again, the programmer must write a local and a remote JCL. The local JCL calls BFXTI and issues a SUBMIT command. The remote JCL is card-images that are the batch job. This remote JCL is encapsulated in the local JCL.

The local JCL is then submitted on the local host for processing by BFXTI. BFXTI sends the remote job over the network to the BFXJS program on the remote host. BFXJS then submits the remote job on the remote host. (The remote host must be able to process batch jobs.)

Remote Job Submission Example

The following is a remote job submission example that is generalized so that the user can understand the general concept of this application. (An actual JCL includes control language statements that are excluded from this example.)

Assume that a user on the local host wants to send job A to the remote host using remote job submission. The user writes the following local JCL:

```
$      IDENT   MYID,SUBMIT
$      USERID  MYID$PASS
$      SELECT  BFX/BFXTI
JOBSUBMIT TO HOSTB JOBFILE=FC*JF
$      DATA    JF,,COPY
$      DUMMY   GENEW
              .
              .
              .
       any jcl
$      ENDJOB
$      ENDCOPY
$      ENDJOB
```

Notice that the remote job is encapsulated in the local job command statements.

Figure 4 on page 9 illustrates the remote job submission process using these sample local and job files.

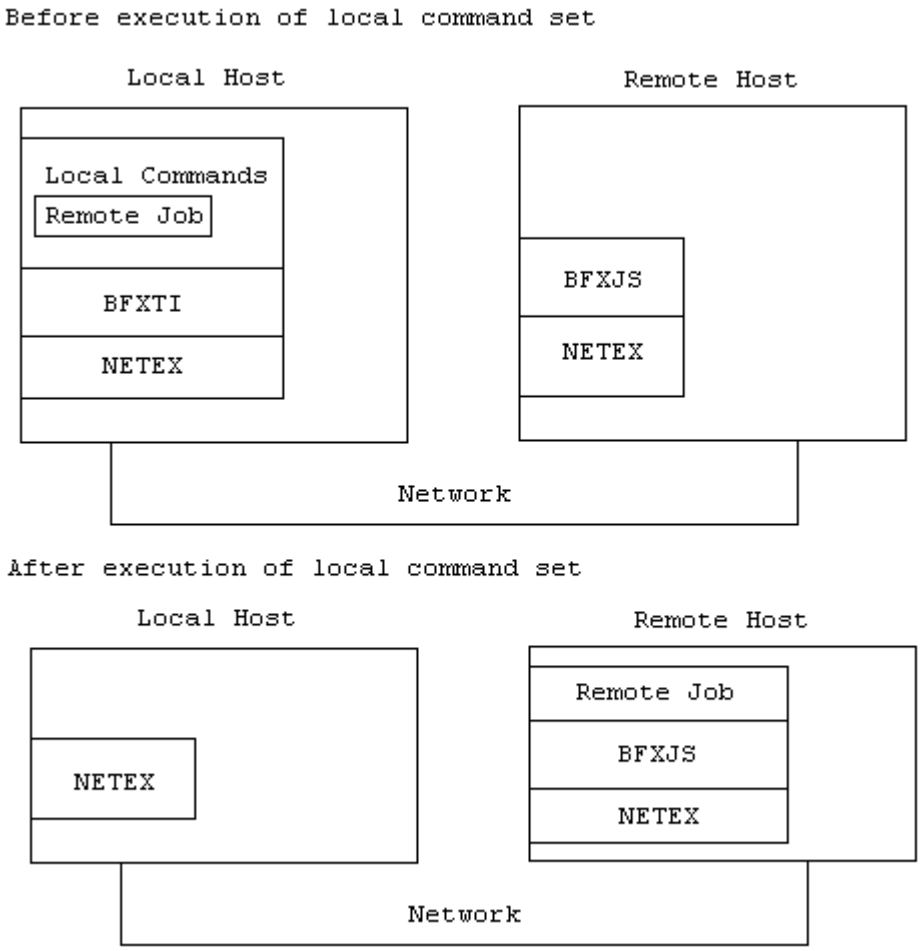

Figure 4. Remote Job Submission Example

Figure 4 shows that BFXTI first scans to see if there is a remote job to be submitted. When the remote job is found, BFXTI establishes a NETEX connection with BFXJS on the remote host. BFXTI then transfers the remote job to the BFXJS on the remote host. BFXTI then terminates. BFXJS submits the remote job for processing on the remote host, then terminates.

Data Modes

The BFX utility transfers the data on a logical record basis using two selectable types of data modes for host-to-host transfers:

- Bit string - a verbatim transfer of a continuous string of bits sent from one host and received as a continuous string of bits by the other host.
- Character - groups of bits (characters) sent from one host and received as groups of bits (characters) by the other host. The number of bits in a group (bits per character) and characters packed per word depends on the character set used and the word size avai1able to each host. Character mode allows most sequential source or text files to be moved between dissimilar hosts in a significant form.

BFX is designed to read a sequential file on the sending host, transfer it to the receiver, and write a sequential file on the receiving host. If the user requires non-sequential operations, encryption, or sophisticated data

conversion, he/she must furnish a user module to perform the operation. For example, with a user module a non-sequential file could be converted to a sequential file (using standard methods), transferred using BFX, and converted back to non-sequential format.

Security

Because BFX is an ordinary batch job to the host, no security mechanisms are required or supplied by BFX. Security mechanisms are not required because the existing host restrictions on utilization, validation, and accounting will remain in effect.

File Size

BFX was intended to be used on files which are greater than one megabyte. The transfer of these large files is highly efficient using BFX. Transferring smaller files is also possible (but not as efficient because elapsed time and batch job queue overhead is relatively higher). Therefore, BFX may be used to satisfy most data transfer needs.

Summary

BFX is a system that consists of three separate programs:

- BFXTI (BFX Transfer Initiate) is an executable image that is started in the computer that wishes to either send or receive a file. Input to BFXTI consists of:
    1. A set of commands that describe the direction of transfer, the process with which transfer will take place, and which user exits (if any) will be used during the transfer process.
    2. A logical file name assignment statement specifying a file to be used for either input or output, depending on the direction of file transfer.
    3. A logical file name assignment statement specifying a file that contains a complete job to be submitted on the partner computer, including all job options, accounting cards, control statements, and the like. This job will invoke the next component, BFXTR, of the BFX system.
- BFXTR (BFX Transfer Responder) is a program that is started in the partner computer,. This program is invoked by the job provided to BFXTI in the initiating process. Its input contains much the same information as BFXTI's, namely:
    1. A set of commands describing the direction of transfer, the node that initiated the transfer process, and a unique ID of the BFXTI program.
    2. The equivalent of a logical file name assignment statement specifying a file to be used for either input or output, depending on the direction of data transfer.
- The last component, BFXJS, is a resident program that must be present in the non-initiating CPU when BFX is to be run. Its sole function is to accept the job destined for the responding host that was provided to the BFXTI program. BFXTI and BFXJS use NETEX to transfer the job, and BFXJS submits the job, unchanged, to the internal reader of the responding machine. It should also be noted that BFXJS can be used independently of the rest of BFX to provide a simple, high speed Remote Job Entry facility.

BFX is principally a batch utility for the transfer of sequential files between computers. There are three basic ways to use BFX:

Automatic Job Submission

The initiating party sends a card image file to the BFXJS job submission program. The BFXJS program submits the file to the batch internal reader. The batch process, once started, invokes BFXTR to connect back to the original initiating party. At that stage a file, or series of files, can be transferred between the initiator and the batch program. This technique is well suited for the transfer of very large files or for the delivery of a file to the initiator.

Remote Job Submission

The initiating party submits a card image file to the BFXJS utility. The associated JCL statements build a file on the remote computer based on the card image data in the job. This is probably the simplest way to "send" a file to another machine.

Manual Job Submission

Two programmers (or operators) on two hosts agree to send a specific series of files in a manually coordinated way. The first party invokes the BFXTI program without submitting a job to BFXJS; the second party then invokes BFXTR to connect to the first party and transfer files.

Operator Interface

This section describes the format and use of commands that execute the BFX programs.

Rules for Coding Commands

Commands are read as a sequence of tokens. Tokens consist of any sequence of characters, delimited by the beginning and end of lines, spaces, equal signs, or commas. Tokens are either considered command tokens or parameter tokens. Command tokens may be at most twelve characters long and must match the name of some BFX command. Command tokens and keyword parameter tokens can be abbreviated if they remain unique. The legitimate values of parameter tokens depend on what command they are parameters for (as described in this section).

To describe a transfer, the programmer must use several commands. A collection of commands used to describe a transfer are said to make up a logical line. A logical line is a sequence of lines where all but the last line have the command token "-". Any data after the command token "-" in a line is ignored, since logically the next line is attached at that point. At most, 160 characters are processed for any input line, and neither leading nor trailing line numbers are allowed.

For example:

```
SEND FILE SAMPLE TO VM4341 ID BFXJOB
```

and

```
SEND FILE = SAMPLE –
TO = VM4341
TO = BFXJOB
```

Both of these logical lines contain four commands used to describe this transfer.

In keywords, all characters are treated as uppercase. In value fields, no translation is performed. Therefore "RPARM HOST1" and "rparm HOST1" are equivalent, but "RPARM HOST1" and "RPARM host1" are not.

Note: The BFX translates all input parameters to upper case. The use of uppercase in the following section (see Figure 5) merely defines the minimum spelling of the keyword.

Figure 5 lists commands and parameters for H291 BFX. The paragraphs following Figure 5 describe the parameters in detail.

General Information about BFX Commands

The order which commands are specified is not important, with the following exceptions:

- If the same parameter appears more than once between two transfer commands, the last specification is used. The same is true for contradictory commands (for example, TIMESTAMP/NOTIMESTAMP).
- If a NOSUBMIT command follows a NEWHOST command, a job file will not be sent if the next transfer statement is SEND or RECEIVE; if NOSUBMIT precedes NEWHOST, however, a job file is sent.
- On BFX JCL cards, line numbers should be stripped before submitting them for batch execution.

Examples of the use of commands are given in "Applications" on page 27.

BFX Command Defaults

Some commands may only be issued to certain BFX programs, such as BFXTI. Table 2 on page summarizes where each command may be issued. Most of the parameters take default values that are in effect at the start of execution of the BFX program (shown in Table 2). Specifying a new value for such a parameter modifies its value for the duration of the run (or until another new value is specified).

Default values are established in module INITROL; recompilation (PASCAL) is necessary to change these values.

BFX Command Summary

The following table summarizes the commands that are available. Detailed descriptions for these commands can be found after the table.

Table 1. BFX Commands

Command	Brief Description	Page
SEnd	Specifies that this logical line describes the transmission of a file to a remote host.	14
REceive	Specifies that this logical line describes the receipt of a file from a remote host.	14
JOBSubmit	Forces a job to be submitted on the remote host.	14
BLock	Size of network block during data file transfer	14
BMod	Module to send/receive blocks	14
BParm	Parameter string for Block Module	15
CIsz	Control interval size of the file	15
DElaytime	Delay time between SCONNs	15
FILe	Logical (or actual) name of data file	15
FIXed	Fixed length records	16
FORMat	Bull file format - GFRC/UFAS	16
FRom and TO	Remote host name	16
Id	BFX application name	16
JBlock	Size of block during job file transfer	16
JCLMode	Character set for job file - ASCII/BCD	17

Table 1. BFX Commands		
Command	**Brief Description**	**Page**
JId	Offered name of BFXJS job	17
JMod	Module to handle remote job submission	17
JOBFile	Logical (or actual) name of job file	17
JParm	Parameter string for Job Module	17
JRmaxl	Maximum record length for the transfer of a job file	18
LName	File name for tape label	18
MEDia	Media code for GFRC format files	18
MOde	Character set or data format type.	18
MSglvl	Minimum severity of messages to be logged	19
NEwhost	New remote host name	20
NOSUbmit	Used when the paired BFXTI and BFXTR jobs will be started independently.	20
NSER	No block serial numbers	20
RMAxl	Maximum record length the file transfer	20
RMOd	Module to read/write file records	20
RParm	Parameter string for Record Module	21
SOe	Stop on Error (SOE) causes BFX to stop the job if a file transfer fails.	21
NOSOe	No Stop on Error (NOSOE) causes BFX to continue with remainder of job even if a file transfer fails.	21
TIMEOFfer	Timeout value for SOFFR	21
TIMEOUt	Timeout value for SREAD	21
TIMEStamp	Print a timestamp with all subsequent BFX messages.	22
NOTimestamp	Suppresses timestamp with all subsequent messasges.	22

BFX Command Descriptions

SEnd

This command specifies that this logical line describes the transmission of a file to a remote host. It is legal in BFXTI and BFXTR. SEND takes no parameters. The minimum spelling for the command is “SE”.

REceive

This command specifies that this logical line describes the receipt of a file from a remote host. It is legal in BFXTI and BFXTR. “RECEIVE” takes no parameters. The minimum spelling for the command is “RE”.

JOBSubmit

This command forces a job to be submitted to the remote host, even if this is an exchange with the same host as the previous logical command line. This command also indicates that no file is to be sent. This command is legal only in BFXTI. JOBSUBMIT takes no parameters. This command is related to the “NOSUBmit” command. The default is “JOBSUBMIT”. The minimum spelling for the command is “JOBS”.

BLock

This command is used to specify the maximum size (in words) of the buffers of data to be sent through NETEX to/from the other host during data file transfer. The block size must be at least large enough to accommodate the largest logical record in the file to be sent or receive

The maximum size allowed is the smaller of 16383 words in CHARACTER mode, 14562 (16383 * 8/9) words in BIT mode, or the NETEX maximum block size as set by the site. The default value for BLOCK is the NETEX default block size as defined by the site. This command is legal in BFXTI and BFXTR. The minimum spelling for the command is “BL”.

For more information on the use of the BLOCK parameter see “Special Considerations” later in section.

BMod

The BMOD user exit block module is for use in building data blocks for transmitting or disassembling received data blocks. To be accessed, references to the Block Module must have been previously added to the NetEx Software-supplied module BFXUIM, and the BFX programs rebuilt.

The BMOD command takes an unabbreviated command token, which is looked up in a list of user exits, as a parameter. If this parameter is not specified, a default Block Module is used. Normally, this will be the NetEx Software-supplied module, which is designed to call a Record Module to obtain or store file information, and to provide or decode the protocol information needed to block and unblock records flowing over the network.

See the “User Modules” section of this manual for more information on user-supplied Block and Record modules.

The Block Module name is an alphanumeric string that is one to eight characters in length. BMOD is valid for all BFX components. The minimum spelling for the command is “BM”.

BParm

The BParm command specifies a string of parameter information that will be passed to the specified Block Module for processing. The parameter is a token (or quoted string) of up to 64 characters. User-written Block Modules may use this string to control special processing options. The default NetEx Software-supplied Block Module ignores the BParm string.

This command is valid in any BFX component. The minimum spelling for the command is “BP”.

CIsz

The CISZ command specifies the control interval size of the send or receive file. The parameter must be a decimal number acceptable to UFAS in the range record size ≤ *nnnn* ≤ 24570. UFAS defaults are used unless a CISZ command is included.

Table 2. CISZ Default Values

Format	Mode	CISZ
GFRC	ASCII	1280
GFRC	BCD	1920
UFAS	ASCII	2048

This command is valid in BFXTI and BFXTR, and applies only in a CHARACTER MODE·(BCD or ASCII) transfer. The minimum spelling for the command is “CI”.

DElaytime

This command specifies how long in seconds the BFX component will delay between reconnect attempts to reach the remote host.

Although legal in all BFX components, it is only effective in BFXTR. This command takes an integer parameter. The default parameter value is 20. The minimum spelling for the command is “DE”.

FILe

The FILE command specifies the name or file code of the file to send or receive. It takes one parameter that is a file name or a file code.

If a file name is specified, it must be contained within 72 characters. The userid may be allowed to default by beginning the name with a “/” (catalog delimiter). The file is dynamically attached and detached. If the output file does not exist, BFX creates a 24 llink file with an unlimited maximum size. Errors encountered in file creation or file allocation (attach) are reported and causes the transfer to terminate. The userid for the activity must have proper permission to create or attach the file.

To specify a file code, the format is FC*xx; where *xx* is the two character file code as defined in the JCL. The file may be a permanent file (PRMFL), temporary disk file, or a tape file. The FC*xx form is the only allowable form for files other than cataloged disk files. Cataloged tape files must be allocated via JCL.

The FILE command is legal in BFXTI and BFXTR. The default parameter value is “FC*TF”. The minimum spelling for the command is “FIL”.

FIXed

The FIXED command specifies that records are to be read or written as fixed length records. It is only valid when MODE is set to one of the character modes (CHARACTER, ASCII, BCD, EBCDIC, or OCTET). If the file does not have fixed length records, a UFAS error is not reported.

The FIXED command is legal in BFXTI and BFXTR. There is no default value. The minimum spelling for the command is “FIX”.

FORMat

The FORMAT command allows specification of a file format. GFRC, UFAS, ANSI, IBM, and HB are allowable formats. ANSI and IBM are allowed only on tape files. If ANSI or IBM is specified for mass storage files, a UFAS error will be reported. HB specifies Bull Bit format and is only used to transfer files using records that are multiples of 64-word sectors.

The default for CHARACTER MODE is GFRC file format.

The default for BIT MODE for disk files is to ignore file format. With MODE BIT and no FORMAT specification, the entire file content is transferred with no regard for end-of-file marks. With MODE BIT and FORMAT GFRC, the file is transferred up to the point that a GFRC EOF mark is found. Note that only one file is transferred with MODE BIT and FORMAT GFRC, so FORMAT GFRC should not be used if a multi-file file is to be transferred. With MODE BIT and FORMAT UFAS, the UFAS file attribute block is used to determine the end of file. With MODE BIT and FORMAT HB, the entire file content is transferred without looking for end-of-file indicators.

The FORMAT command is legal in BFXTI and BFXTR. The default parameter value is “GFRC”. The minimum spelling for the command is “FORM”.

FRom and TO

These commands specify the host with whom a file is exchanged. FROM and TO are logically equivalent. The two forms are provided for readability only. FROM and TO accept one parameter which is a one to eight character NETEX host name.

FROM and TO are legal in BFXTI and BFXTR. The default parameter value is “LOOPBAK”. The minimum spelling for the commands are “FR” and “TO”.

Id

This command specifies a unique name by which the initiator (BFXTI) and responder (BFXTR) will identify themselves to each other. The “ID” command is required for exchanging files. Only one BFX program with this ID should be present in either the local or remote host. The ID parameters for BFXTI and BFXTR must be the same, or the file transfer will fail.

ID is legal in BFXTI and BFXTR. This command takes one parameter, the first eight characters of which are significant. There is no default parameter value. The minimum spelling for the command is “I”.

JBlock

This command specifies the block size to use in exchanging the job file with the remote host. This command takes one numeric parameter. The maximum size allowed is the same as the BLOCK command.

The JBLOCK command is legal in BFXTI and BFXJS. The default parameter value is 1024. The minimum spelling for the command is “JB”.

JCLMode

This command specifies the character set for the job file to be sent to the remote host. The value BCD indicates the job file is stored in the BCD character set. The value ASCII indicates the job file is stored as a media code 6 ASCII file (standard TSS format).

The JCLMODE command is legal in BFXTI. The default parameter value is BCD. The minimum spelling for the command is “JCLM”.

JId

This command specifies the id of the BFX component to exchange job files with. This command is legal in BFXTI. The parameter is an application name, like the ID command parameter. The default value is BFXJS. Normally, this need not be changed unless multiple copies of the BFXJS program are in use. The minimum spelling for the command is “JI”.

JMod

This command specifies the name of the user exit module (the Job Submission Module) that will take responsibility for the submission of the job file to BFXJS on the remote host. To be accessed, references to the Job Submission Module must have been previously added to the NetEx Software-supplied module BFXUIM, and the BFX programs rebuilt.

If this parameter is not specified, a default Job Submission Module is used. Normally, this will be the NetEx Software-supplied module, which uses the NetEx Software-supplied Block and Record Modules to transfer character files between all types of host computers.

See “User Modules” on page 43 for more information on user-supplied Job Submission modules.

This command is valid only in BFXTI. The parameter is an unabbreviated command token which is looked up in the list of user exit modules. The default parameter value is JOBSTANDARD. The minimum spelling for the command is “JM”.

JOBFile

This command specifies the name of the job file to send to the remote host. This file is sent when switching to a new host or the JOBSUBMIT command is given.

This command is only legal in BFXTI. It takes one parameter, which is a file name or file code. Refer to the FILE command. The minimum spelling for the command is “JOBF”.

JParm

This command specifies a string of parameter information that will be passed to the specified Job Submission Module for processing. User-written Job Submission Modules may use this string to control special processing options. H291 BFX uses the JParm parameter to define special transliteration of JCL files. Refer to H291 BFX Implementation Notes on page 31.

The parameter is a token (or quoted string) of up to 64 characters. JParm is legal only in BFXTI. The minimum spelling for the command is “JP”.

JRmaxl

This command specifies the maximum record length for the transfer of a job file. The parameter specified with the JRmaxl command is a numeric token.

The JRmaxl command is legal in BFXTI and BFXJS. It accepts an integer value for a value. The default is 80. The minimum spelling for the command is “JR.”

LName

This command specifies the file name to be placed into the label record of a magnetic tape. UFAS uses the first 17 characters for labeled UFF, ANSI, and IBM files and the first 12 characters for labeled GFRC files.

The LName command is legal in BFXTI and BFXTR. There is no default value for this command. The minimum spelling for the command is “LN.”

MEDia

The MEDIA command specifies the record media code for GFRC format files. The parameter must be an integer number of a valid media code 0-13.

```
Defaults: ASCII = 6 (TSS ASCII)
          BCD   = 2 (CARD FORMAT)
```

This command is valid in BFXTI and BFXTR, and applies only to a CHARACTER MODE (BCD or ASCII) transfer. Users interested in the default value should refer to the UFAS manual. The minimum spelling for this command is “MED.”

Note: For BCD records, if the length is variable, or greater than 80 characters, MEDIA 0 should be specified.

MOde

The MODE command specifies the character set or data format for the file exchange. The valid commands for character mode are listed below:

- CHaracter (default, equivalent to ASCII)
- Ascii (ASCII character set)
- BCd (Bull 6-bit character set)
- Ebcdic (EBCDIC character set)
- Octet (8 bit characters, media code is ignored)

The valid commands for binary transfers are listed below:

- BIT (no file formatting recognized)
- BINaryrecord (transfer contents of each UFAS binary record in binary mode)

In the character modes (CHaracter, Ascii, BCd, and Ebcdic), the text is converted from the defined character set of the sending machine to the receiving machine. The logical records of the character information are converted to the logical record structure of the receiving host.

In BIT mode, the exact pattern of bits in files are sent to and stored by the receiving host. This includes block control words, records control words, or other data. On some other host systems, BFX includes additional control information to files received in BIT mode. When sending files in BIT mode, the BFX RMAXL pa-

rameter defines the number of words sent in each record to the destination system, regardless of any control information contained in the file.

In BINARYRECORD mode, UFAS or writes each logical record for BFX. On sends, each logical record is transmitted as a single BFX record, giving the length in bits of the logical record. On receives, BFX takes the length in bits of the received record and writes a single UFAS binary record. The record length may be increased by up to 35 bits to form an integer number of words in the record. Often a file transferred to a GCOS8 system in BINARYRECORD mode, and then sent back to the originating system is larger because of these added bits. BINARYRECORD mode is consistent with BIT modes on other host systems.

Choosing between BIT and BINARYRECORD modes for binary transfers requires an understanding of both sending and receiving system file formats and BFX options. The choice may be based on some of the following factors:

- BINARYRECORD mode uses UFAS to extract or write logical records. BIT mode extracts or writes data based on the record lengths being sent or received. Between GCOS8 and other types of systems, the recommended transfer mode depends on the application. If the data is simply being moved to another system to be stored for later retrieval, BIT mode is the appropriate mode. If the remote machine has routines to process GCOS8-formatted data including UFAS or GFRC control words, BIT mode is appropriate. If the data is to be interpreted directly by an application on the remote machine, BINARYRECORD is likely to be appropriate.
- Very few applications between unlike hosts use either BIT or BINARYRECORD mode. Between unlike hosts, a CHARACTER mode is normally appropriate. Between GCOS8 machines, either BIT (combined with FORMAT HB) or CHARACTER modes should be normally used.
- BIT mode and BINARYRECORD mode are inconsistent with CHARACTER mode.

The MODE command is legal in BFXTI and BFXTR. The default value for the command is “CHARACTER”. The minimum spelling of the command is “MO.”

MSglvl

This command specifies which BFX messages will be displayed. The parameter must be specified as an integer value in the range 0-16. Messages with severity levels greater than or equal to the specified value are written to OUTPUT. Message level 16 is not recommended for use. Setting the message level to 16 will inhibit all error messages. The meanings of the various message levels are shown below:

- 0-3 Messages that are generated for diagnostic purposes. These messages will trace the flow of events in detail, and are intended only for use in diagnosing problems with BFX, NETEX, or newly-written user modules.
- 4-7 Messages indicating the status of job submission, starting of the remote BFXTR job if applicable, and statistics on the file transferred.
- 8-11 If transfer of the file is normal, this will generate only the “transfer complete” message. If an error that causes the run to finish is encountered, then the error messages will be logged.
- 12-15 Only errors that cause the file transfer process to be aborted will be printed.
- 16 Inhibits the output of all messages.

This command can be issued in any component. The default value for this command is 4. The minimum spelling for this command is “MS.”

NEwhost

This command specifies a new host to exchange files with after one or more transactions have been performed with another host or hosts. It is effectively equivalent to the “TO” and “FROM” commands. NEWHOST takes one parameter which is the 1 to 8 character NETEX hostname.

This command is legal only in BFXTI. There is no default value for this command. The minimum spelling of this command is “NE.”

NOSUbmit

NOSUBMIT is used in cases where each half of the BFX pair will be started independently on the two hosts. In such cases, the BFXTI job with the NOSUbmit statement should be started first; the BFXTR job should be started after BFXTI is offered. If this command is specified, BFXTI will not send a batch job through to BFXJS on that host.

The NOSUBMIT command is legal only in BFXTI. The default is JOBSUBMIT. It takes no parameters. The minimum spelling for the command is “NOSU.”

NSER

The NSER command specifies that no block serial numbers are present (for a file being read), or no block serial numbers are to be written for an output file. It is useful only in CHARACTER modes, not BIT mode. If NSER is specified for an input file that actually has block serial numbers, a UFAS error will be reported.

The NSER command is legal in BFXTI and BFXTR. The command accepts no arguments. The minimum spelling of the command is “NSER.”

RMAxl

This specifies the maximum record length for the file transfer. The parameter for this is a numeric token. In BIT MODE the parameter specifies a word length. In CHARACTER MODE the parameter specifies the appropriate character length for the code set. The maximum record length that may be specified is 24570 for CHARACTER mode and 14,561 for BIT mode. For BIT mode, RMAXL values in multiples of 64 words are recommended for SEND commands.

This command is legal in all BFX components. The default parameter value for this command is 80. The minimum spelling for this command is “RMA.”

RMOd

The RMOD command specifies the name of the user exit record module that will provide or accept the logical records of the file. To be accessed, references to the record module must have been previously added to the supplied module BFXUIM, and the BFX programs rebuilt.

If this command is not specified, a default record module is used. Normally, this will be the supplied module, which is designed to transfer character files between all types of host computers, and to move binary or structured information to another host without change on a logical record basis.

See “User Modules” on page 43 for more information on user-supplied block and record modules.

The record module name is parameter like a command token except that abbreviation is not allowed. The name is looked up in the list of user exit record modules.

The command is legal in all BFX components. The default parameter value for this command is STANDARD. The minimum spelling for this command is “RMO.”

RParm

This command specifies a string of parameter information that will be passed to the specified record module for processing. User-written record modules may use this string to control special processing options. H291 BFX uses the RPARM parameter to define special transliteration when sending data files. Refer to “H291 BFX Implementation Notes” on page 31.

The parameter is a token (or quoted string) of up to 64 characters. RPARM is legal in all BFX components. The command has no default value. The minimum spelling of the command is “RP.”

SOe

The Stop On Error (SOE) command causes BFX to stop reading further commands if any one transmission has problems.

This command is legal in BFXTI and BFXTR. It takes no parameters. The minimum spelling of the command is “SO.”

NOSOe

The No Stop on Error (NOSOE) command reverses the effects of a previous SOE. It causes BFX to keep reading logical commands even if one transfer fails. “NOSOE” is the default behavior if neither “SOE” nor “NOSOE” are defined in the BFX job.

This command is legal in BFXTI and BFXTR. It takes no parameters. The minimum spelling of the command is “NOSO.”

TIMEOFfer

The TIMEOFFER command specifies the maximum amount of time (in seconds) that BFXTI will wait for the BFXTR job to be initated in the remote host. BFXTI “offers” itself through NETEX to the remote job. If the remote job does not respond within the time allotted by TIMEOFFER, BFXTI will abort the transfer process.

This parameter can be defaulted. It is legal in all BFX components, although it currently only has effect in BFXTI and BFXJS. The default values for this command are 240 (for BFXTI) and 3600 (for BFXJS). The minimum spelling of the command is “TIMEOF.”

TIMEOUt

The TIMOUT command specifies the maximum amount of time (in seconds) that the BFX program should wait for a read through NETEX to complete.

This command should be specified when sending data over low speed links (such as phone lines).

It should also be used by the receiving BFX when there is a possibility that the sending BFX will be experiencing long delays during file transfer. For example, if a multi-volume tape file is being sent, the sending BFX will be delayed when one reel ends and the next reel is being mounted. In such cases, TIMEOUT should be set to a very high value or to 0 (timeout disabled).

The command is legal in all BFX components. The default value for this command is 60. The minimum spelling for the command is “TIMOU.”

TIMEStamp

The TIMESTAMP command specifies that a timestamp is to be printed with all subsequent BFX messages. The timestamp will give the time of day that the message was sent to the print file in the format hh:mm:ss.ttt on the left part of the message.

This command is legal in all BFX components. The default is NOTIMESTAMP. The minimum spelling for the command is “TIMES.”

NOTimestamp

The NOTIMESTAMP command reverses the effect of the TIMESTAMP command. NOTIMESTAMP specifies that no timestamp is to be printed with all subsequent BFX messages. “NOTIMESTAMP” is the default behavior if neither “TIMESTAMP” nor “NOTIMESTAMP” are defined in the BFX job.

This command is legal in all BFX components and takes no parameters. The minimum spelling for the command is “NOT.”

Special Considerations

Transfer to Non-GCOS Computer Systems

The code supplied by NetEx Software is designed to support transfer of file data between “incompatible” computers in two ways (selected by the MODE command):

- Files containing only character information (program source files, text, line printer output) will be converted to an immediately useful form when sent to a different computer.
- Files containing binary information, floating point numbers, or data structures will be sent to the other computer as a continuous string of bits on a record basis. Depending on the types of computer systems involved, the data may be ready for direct use, or some processing of the data may be needed following the BFX run before it is ready for direct use.

Note: BFX does not convert tab characters to blanks and vice versa. If files are sent to a system that does not use the tab character to produce “white space” on a listing, then the tabs must be removed before or after the file is sent.

Use of the BLOCK Command

The value specified for the BLOCK command (or the default value) is not necessarily the block size that is used during file transfer. The minimum of the block sizes requested by the partners in a transfer is the size actually used. However, the size requested by each side can be greater than was specified in that side’s BLOCK command.

It is also important to note that each record sent between the BFX programs has overhead associated with it. This overhead will be a minimum of 6 bytes for bit mode transfers or 8 bytes for character mode transfer; however, it can be higher when transferring between machines with unusual word sizes. The BFX programs do not know exactly how long the overhead will be when they present their desired block sizes. Therefore, BFX assumes a worst case of 21 bytes when figuring overhead into its requested block size.

Applications

Applications of BFX

BFX transfers files from one system to another. This file transfer facility can be combined with other activities to improve processing in environments with multiple systems.

BFX can be used for file backup. Strategic files can be moved to another system to act as on-line files, or they could be transferred to a system that has better tape handling facilities.

Processing systems can be built using BFX to send files containing data such as packing slips or invoices to other applications which use these files as input data.

The job submission functions of BFX can be used to do work entirely unrelated to file transfers from a remote system.

In cases where one system has better editing capabilities, a programmer might edit files on one system and then use BFX to transfer the edited file to an appropriate system.

BFX may be used to distribute program updates in either source or object forms to other systems on a network. Distribution in object forms (object decks, H*'s, and so on) can be done using BIT mode transfers.

Examples

Examples of BFX activities are provided below. These examples assume some knowledge of Bull JCL.

BFXJS JCL

BFXJS is generally run as a job spawned from the OPNSUTIL USERID. The JCL for running BFX in this fashion is shown below:

```
$      SNUMB   BFXJS,68       **Note high urgency**
$      IDENT   OPNSUTIL
$      SELECT  BFX/BFXJS
TIMESTAMP
```

It is possible to run multiple copies of BFXJS on a system. If done for test purposes, copies other than the production BFXJS should include a JID parameter and BFXTI activities wanting to submit jobs should include the same JID parameter. A sample JCL for such a second copy might be:

```
$      SNUMB   BFXJT
$      IDENT   YOURID
$      USERID  MY$ID
$      SELECT  BFX/BFXJS
TIMESTAMP JID BFXJT
```

The following examples assume that BFXJS is running on the remote system.

File Transfer to Another GCOS8 System

The file on the local system is a GFRC Media 6 format file and on the receiving system it is to be stored as a UFF sequential file.

```
$      SNUMB   12345
$      IDENT   M5432,BIN-57
$      PROGRAM BFXTI
$      USERID  MY$ID
$      SELECT  BFX/BFXTI
SEND TO HOSTB ID=MYFLRUN1 FORMAT=GFRC MEDIA=6 –
FILE=MY/INPUT/FILE
$      DATA    JF,COPY
$      DUMMY   GENEW
$      IDENT   M1234,BIN-18
$      USERID  YOURS$TRULY
$      SELECT  BFX/BFXTR
RECEIVE FROM HOSTA ID=MYFLRUN1 FORMAT=UFAS –
FILE=YOURS/TO-READ TIMESTAMP MSGLVL=2
$       ENDCOPY
```

Obtaining a File From Another GCOS8 System

The file on the remote system is an ASCII print image file. To obtain the file on our local system in the same format, the following job could be used:

```
$      IDENT   ACCTN,NAME
$      USERID  MY$ID
$      SELECT  BFX/BFXTI
RECEIVE FROM HOSTB ID=APPL4AT2 FORMAT=GFRC MEDIA=13 MODE ASCII –
FILE=MY/OUTPUT/PRINT/FILE4AT2 TIMESTAMP
$      DATA    JF,COPY
$      DUMMY   GENEW
$      IDENT   M1234,BIN-18
$      USERID  YOURS$TRULY
$      SELECT  BFX/BFXTR
SEND TO HOSTA ID=APPL4AT2 FORMAT=GFRC MEDIA 13 MODE ASCII –
FILE=APPL4AT/OUTPUT2/TODAY
$      ENDCOPY
```

File Transfer to an IBM MVS System

The file on the GCOS8 system is a BCD card image file. The following JCL could transfer the file to an MVS file:

```
$      IDENT   M1234,BIN-18
$      USERID  MY$ID
$      SELECT  BFX/BFXTI
send to mvssys  id=abcdefgh format=gfrc media=2 mode=bcd
$      PRMFL   TR,R,S,MY/BCDFILE
$      DATA    JF,COPY
//L123456  JOB
//SEND     EXEC PGM=BFXTR
//STEPLIB  DD   DSN=BFXL.BFXLOAD,DISP=SHR
//         DD   DSN=NTXL21.NTXLOAD,DISP=SHR
//         DD   DSN=NTXL21.NTXFLOAD,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//TSTFILE  DD   DSN=&&TESTFL,DISP=(NEW,PASS),UNIT=DISK,
//              DCB=SYS.STANDARD.COBOL,SPACE=(CYL,(3,3))
//SYSIN    DD   *
RECEIVE FROM HOSTA ID ABCDEFGH MODE CHAR OUTDD TSTFILE TIME
/*
$      ENDCOPY JF
```

Obtaining a File from a VMS System

The following JCL could retrieve a test file from a VAX VMS system:

```
$      IDENT   M1234,BIN-18
$      USERID  MY$ID
$      SELECT  BFX/BFXTI
RECEIVE FROM vaxsys id=12345 format=gfrc media=6 mode=ascii
$      PRMFL   TR,W,S,MY/OUTPUTFILE
$      DATA    JF,COPY
$ JOB MYVAX
$ PASSWORD SECRETCODE
$ DEFINE SYS$PRINT NL
$ DEFINE TRFILE PSOURCE.DAT
$ RUN BFX_ROOT BFXTR
ID 12345
BLOCK 4096
FILE TRFILE
HOST HOSTA
MODE CHAR
SEND
$ EOJ
$      ENDCOPY
```

H291 BFX Implementation Notes

The following items describe functionality and limitations distinctive to H291 GCOS8 BFX.

File Formats

Several file formats are supported by BFX. BFX uses UFAS for handling character mode I/O. Any UFAS limitations thus become BFX limitations. Certain UFAS options are not supported by BFX.

File types explicitly supported by BFX for character mode transfers include:

```
Disk and      GFRC  Media code 0       (BCD, Variable-Length Records)
tape files    GFRC  Media code 2       (BCD, card image)
              GFRC  Media code 3       (BCD, print line image)
              GFRC  Media code 6       (TSS ASCII)
              GFRC  Media code 7       (ASCII, print line image)
              GFRC  Media code 10      (ASCII, card image)
              UFAS  UFF Sequential Files

Tape files    ANSI
              IBM
```

File types not explicitly supported by this release of BFX include:

```
H2000 Files (disk and tape)   APL Random
UFF Relative Files            APL Workspace
UFF Indexed Files             Huffman Encoded Files
ISP Files                     Mailbox Files
IDS-I Files                   Object Files
DM-IV Files                   Q* Files
Random Libraries              Random Source Libraries
```

For unsupported disk file formats, BIT mode transfers can be used between GCOS systems. User block and/or record modules would normally be needed to transfer these files to or from systems with different disk structures.

If a file type is not listed as being supported, BFX probably does not support the file type. Sometimes UFAS can read the file; in that case, BFX may be able to send the file. NetEx Software, Inc. makes no commitment to support all possible file types.

BIT Mode

When transferring disk files in BIT mode, the entire file content is transferred. This means that all file attribute records, block control words and record control words are transferred which results in an identical copy of the original file.

Tape files may be transferred using BINARYRECORD mode. This transfers the data records of the tape as a continuous stream of bits. Tape labels (including user labels) are not transmitted by BFX. UFAS tape label processing is used so label information depends on prior tape labels, any JCL affecting tape labels, and BFX options (ANSI, IBM, GFRC, UFAS, and so on) that may affect tape labeling.

UFAS Errors

All UFAS errors are reported by UFAS in the execution report in the normal manner.

Record Truncation

When reading GFRC files (only), if the actual record length is larger than the RMAXL command record length, the record will be truncated and a warning message will be reported. This is caused by UFAS truncating the input record for GFRC files to the record length you request. For UFAS files the open will abort if the actual file attributes do not conform to those specified.

Job Spawning

GCOS requires that the first card in a spawned file must be `$ DUMMY GENEW`. A valid userid and password must also be included in the JCL. For GCOS8 users, a job file may be submitted with a `$ SNUMB sssss` card as the first card. BXFJS will use *sssss* as the SNUMB for the spawned job and replaces the `$ SNUMB` card with the required `$ DUMMY GENEW`.

Performance

Performance of BFX is dependent on many factors. Improving performance can sometimes be accomplished by changing BFX parameters.

Generally, larger data block sizes result in faster transfers.

Data rates typically increase as record lengths increase. For character transfers, this is largely a function of UFAS. For bit mode transfers not using the FORMAT HB option, this is not a major factor. For bit mode transfers using the FORMAT HB option, increasing record lengths directly reduces disk I/O as exactly one disk I/O is done per record from the viewpoint of BFX. When using FORMAT HB, record lengths in the range 4K-14K words can provide excellent performance.

When using FORMAT HB, large disk record lengths, and large blocks (4K-14K), disk I/O can become the limiting factor, particularly when using older disk devices

Though FORMAT HB is intended primarily for transfer between Bull systems, it can be used to transfer data to or from any systems that can readily handle data blocks that are multiples of 64 36-bit words. For record sizes over 2K words, FORMAT HB will generally provide better performance than a BIT transfer using the default mode. For record sizes under 1K, FORMAT HB will often be slower than a normal BIT transfer.

Transfer of NULL Files

When a file is dynamically allocated, that is, FILE= < FMS pathname>, a unique FMS status is returned if the file has never been written. When this occurs, an end-of-file .condition will be returned on the first read of a logical record, resulting in a valid transfer of the NULL file.

Data Mode vs. Media Code

This applies only to character mode transfers of GFRC format files. When the first logical record is obtained, the media code found in the Record Control Word (RCW) of the record is interpreted and matched against the

data mode of the transfer. If the data mode indicates BCD data, the media code should be accepted as BCD (as described in UFAS documentation). Conversely, if the data mode is ASCII (default), the media code should be compatible. If a mismatch occurs, the status will be reported using the BFXI02 error message (see the following description), and the transfer will be terminated.

Transliteration

Beside the standard transliteration tables provided by NETEX, H291 BFX provides the capability to transliterate data files before transmission across the NETEX network.

This option is available only when SENDING files; no transliteration is done by BFX for data received across the network.

Only the data file content is transliterated; the standard ASCII character set is used for BFX messages.

This code option provides for transliteration from 6-bit to 8-bit, or 8-bit to 8-bit, character sets; 6-bit to 6-bit is NOT supported.

The BFX "RPARM" command is used to define this option. As defined in the "Operator Interface" on page 13, this parameter must be either a token or a quoted string, so for our purposes this takes one of the following forms:

- `RPARM "XLIT = xxxxxx"`

  xxxxxx

  This is the transliteration table name to be loaded.

- `RPARM NOXLIT`

  This command turns off a previously defined transliteration.

In addition, the same parameter format may be used with "JPARM" to invoke transliteration on JCL files to be sent to BFXJS on another system.

Note: Use of JPARM does NOT imply transliteration of DATA flies; conversely, use of RPARM does not imply transliteration of JCL files; BOTH parameters must be used if both JCL and data are to be transliterated, even if the same transliteration table is to be used.

The equals '=', and spaces ' ', are optional, but at least one of these delimiters must separate each keyword or parameter.

The RPARM parameter needs to defined only once for multiple file transfers. When a transliteration table is invoked, it will remain in effect for all following files until another transliteration table, or option, is selected. This also applies to JCL files when the JPARM parameter is used.

The default transliteration file name is **BFX/XLIT/XLIT.H**, a multi-element H* (O*) file. The file must be allocated using a "$ PRMFL JCL" statement, using the file code "XT".

If, in the process of loading or validation of the selected transliteration table, an error occurs, the standard BFX message will be generated. The message is shown below:

BFX210S CANNOT OPEN INPUT FILE fffffff. RC=ccc.

Besides the codes that may be returned by UFAS or FMS procedures, the error code "ccc" may contain the following:

1991 Invalid table size. The transliteration table elements in the file must be EXACTLY 16 words for 6-bit (BCD) to 8-bit (ASCII, EBCDIC, ...) transliteration, and either 64 or 128 words for 8-bit to 8-bit transliteration.

1992 Invalid (JPARM) parameter. Something other than the keywords “XLIT” or “NOXLIT” was included, or some other error such as a missing quote, occurred during parsing.

1993 Invalid character in table. Using a character value greater than 377 (octal) for a character mode transfer will result in data parity errors from the system I/O processor (IOP, IOX, ...), after which NETEX (H290) will “and off” the high-order bits and retry the transmit (H297 NETEX will return an error and close the network connection). Since this could cause system degradation, and since users may define their own transliteration tables, BFX will validate each table that is loaded to ensure that all characters are in the range 000 to 377 (octal). An invalid entry will result in an error condition, and the file transfer(s) will not be done.

1994 This code indicates a mode/format incompatibility when a transliteration table is present. It may indicate that FORMAT is not GFRC or UFAS (for example, IBM or ANSI are invalid); or that the output MODE is not one of ASCII, EBCDIC, or OCTET. This code is used if the transliteration table does not appear “correct”. For example, if the table is 16 words long, and MODE = ASCII, the ‘0’ (zero) character position of the table must an ASCII ‘0’ (octal 060); if the table is 16 words long and MODE = EBCDIC, the zero position must be an EBCDIC ‘0’ (octal 360). If the table is either 64 or 128 words long, it is assumed to be an ASCII to <8-bit> transliteration, and the table entry for the ‘0’ character (byte offset 48 decimal or 060 octal) must also be valid for the output MODE (octal 060 for ASCII, octal 360 for EBCDIC).

Note: These tests will not be made if MODE = OCTET

1xxx This range of error codes indicates that an error was returned from the GCOS8 MME GERSTR processor; xxx indicates the code that was returned. For example, RC= 1063 indicates a “call name missing” error, probably because of a misspelled name in the XLIT parameter.

An additional test is made when the first data record is obtained from the file. As described above in “Data Mode vs. Media Code” on page 33, BFX includes code that attempts validation of data mode versus the media code of the file. When data transliteration is invoked, additional tests are made; in the previous example, the BCD media to ASCII data mode will be allowed if a 6-bit (BCD) to ASCII transliteration table is selected, and BCD to EBCDIC will be allowed if a BCD to EBCDIC table is selected. If the media code indicates BCD data, then the transliteration table, if present, must be 16 words in length; if the media code indicates ASCII data, the table must be either 64 or 128 words. If this test fails, the BFX102S message will be generated, coded as described in the following paragraphs.

BFX102S Message Notes

The BFX102S (“File <filename> PERMANENT I/O ERROR - - RC = < nnnnn>”) is a generic BFX message generated whenever the record processing module encounters an error. To differentiate between codes returned from the various system elements (UFAS, FMS, internal, ...), some of the codes are changed as follows:

- FMS allocation errors are reported as (8000 + code); for example, a “permissions denied” error would be displayed as “RC = 8003”.
- Media code/data mode mismatches are The negative of “(data mode times 100) plus media code”. For example, a BCD media code= 3 and an ASCII data mode = 2 without a valid BCD to ASCII transliteration table would result in RC = -203. The media codes used by GCOS8 are defined in various documents available from Bull such as DH07 and DH40 (UFAS Reference Manuals); some of the more

prevalent codes are: 2= BCD card image, 3=BCD print, 6=TSS ASCII, 7=ASCII print. The codes used by BFX for data mode are: 0= BIT, 1= OCTET, 2=ASCII, 3= EBCDIC, 5= BCD.

- UFAS errors are displayed as returned from the library routines.

Installation

This procedure assumes that the installation is performed by a systems administrator or programmer familiar with the GCOS8 file system, JCL, and software necessary for installation.

Topics discussed in this section are:

- Prerequisites
- Installation of BFX
- Installation of the transliteration option
- Updating transliteration flies
- Updating PFX

Prerequisites

The following prerequisites must be met for a successful H291 BFX installation:

- A Bull GCOS8 system with an operational TCP/IP networking configuration.
- A currently supported release of H297IP NETEX/IP Requester.
- A NetEx Software TNP Product (i.e. TNP210).
- At least one other host with an appropriate hardware and software configuration.
- An IP network connecting the two (or more) hosts.
- If BFX is to be customized, a currently supported version (PCS1.2) of the PASCAL compiler including all updates, installed in the time-sharing library (CMDLIB).

Installation Procedures

The following tasks are necessary to make BFX usable on the GCOS8 system.

1. Download the software, delivered as a single-file User Save.
2. Restore the save.
3. Install (optional) transliteration flies.
4. Test BFX.
5. Install JCL files.

Step 1. Download software

Download the software to the file BFX/BFX3RELEASE file.

Step 2. Restore the Save Tape

H291 is supplied on a file in the file system user save format. Refer to the H291 BFX3.x Memo to Users for the list of files contained on the tape, and the space requirements.

Note: If a System Master Catalog (SMC) entry for BFX does not already exist, it must be created. The space required for all BFX3.x files is specified in the Memo to Users. Other parameters such as time-sharing resources and permissions are not required, so are at the discretion of the site administrator. If the BFX userid already exists, it may be necessary to increase the maximum space allowed to meet the space requirements.

A sample FILSYS restore JCL setup follows:

```
$      ident   <required fileids>
$      filsys
$      privity
$      prmfl   pr,r,s,bfx/bfx3release
userid bfx
restore bfx/bfx3.x
$      endjob
```

Files under directories such as TOOLS and TEST may prove useful during installation and initial testing, and then may be released at the sites discretion.

After the restore is complete, a check should be made for the presence of file **BFX/BFX3.x/README**. If present, this ASCII text file may contain miscellaneous errata, revised installation instructions, and so on, that should be reviewed before proceeding with the installation.

Step 3. Update the IDENT File

The file BFX/BFX3.x/IDENT should be created to reflect local requirements, including account numbers, delivery stations, and so on. This file should be saved in CARD format so it can be referenced in batch activities.

Because of the site-specific nature of IDENT cards, NetEx Software cannot describe in detail all changes that may need to be made.

For example:

```
$      IDENT   xxxx,NETEX
```

Step 4. Test BFX

A CRUN is delivered under the BFX/BFX3.x/TEXT directory for testing. The CRUN clears an output file, initiates a BFXTI job to send a file, and initiates a BFXTR job to receive a file. These two jobs should run without change. The file BFX/BFX3.x/TEST/TESTRECV should contain the same data as the file BFX/BFX3.x/TEST/TESTFILE after the jobs are completed.

H297IP users should enter the, following command:

```
CRUN BFX/BFX3.x/TEST/CRUN297IP
```

If the jobs do not run, or if the data in file BFX/BFX3.x/TEST/TESTRECV appears incorrect, any messages contained on the execution reports for the jobs (Report Code $$) or the output report for BFX (Report Code 72) should be investigated. A copy of normal output for these runs is in file BFX/BFX3.x/TEST/NORMAL.

Step 5. Install JCL Files

Sample JCL is included on the distribution tape under directory **BFX/BFX3.x/JCL**. It is recommended to generate a spawn file under OPNSUTIL. CRUNs are provided to create a spawn file and set up SELECT files **BFX/BFXTI** and **BFX/BFXTR**. Users of earlier BFX releases should review these CRUNs to determine the proper time for executing them.

Enter the following command:

```
CRUN BFX/BFX3.x/INSTALL/IF.H297IP
```

Commentary is included in the CRUN flies and in the Memo to Users to assist in the installation.

BFX should now be installed and BFXJS can now be SPAWNED from the operator's console.

Notes on JCL for the Transliteration Option

Since NETEX provides for character data transliteration, most sites should skip this installation step. Special transliteration is an option provided for those sites that have requirements, such as handling EBCDIC characters not handled by NETEX, foreign character sets for specific applications, or file transfers to specific systems having non-standard character sets. This option is primarily useful if a site needs different transliteration tables for different applications.

After the above JCL installation, file **BFX/BFXTI** (or **BFX/BFXTR**) would normally contain JCL statements such as the following:

```
$      program   bfxti (or bfxtr)
$      limits    ,64K
$      prmfl     **,r,r,bfx/BFX3.x/run/hstar
```

If most of the sites users use the normal procedure for selecting this JCL file to invoke BFX, the easiest method of providing these users with the transliteration capability would be to add the following JCL statement to these two JCL files:

```
$      prmfl   xt,r/c,r,bfx/xlit/xlit.h
```

Given the above, end users will then only need to include the appropriate RPARM options in their JCL.

If there are several high use procedures that involve using BFX, there are variations of the above example that can be used to make the end users task less tedious. As an example, suppose there are many activities involved in transferring BCD data files to an IBM system, and the data mode desired at the receiving end is EBCDIC. A JCL file can be set up as follows (our example uses the name **BFX/BCD2IBM**):

```
$      PROGRAM BFXTI
$      LIMITS ,48K,,lK
$      prmfl   **,r,r,bfx/BFX3.x/run/hstar
$      prmfl   xt,r/c,r,bfx/xlit/xlit.h
send to ibm sys -
rparm = "xlit bcdebc"
mode ebcdic -
block 12288 -
```

The user may now generate JCL as follows (assuming proper USERID, IDENT, ...):

```
$     select bfx/bcd2ibm
id sendit -
nosubmit -
block 1024 -
file my_id/my_bcdfile
```

As long as the users BFX commands IMMEDIATELY follow the "$ SELECT" statement, the GCOS8 system input processor (GEIN) will concatenate them following BFX commands included in the selected file. There must not be any intervening JCL statement ($ in position 1). Since BFX, in most cases, does not require commands to be in any particular order, the above example results in a valid BFX command "line". Note the presence of the BLOCK command in both the selected JCL and the user's parameters; the second (user's) specification would be the one used.

Of course, if most jobs of this type are submitted via time-sharing, the site may wish to set up ASCII JCL files and instruct users to use the TSS selection method to accomplish the same result, for example:

```
$$select(bfx/bcd2ibm)
id sendit -
nosubmit -
block 1024 -
file my_id/my_bcdfile
```

Installing H291 Updates

New releases to the H291 BFX software product will be distributed via the H291 Distribution in the same format as the original release, except that the second level catalog name, for example, BFX3.0, will change. Therefore, installation will be identical with the detailed process described above, except for the task related to creating the BFX SMC entry. (It may be necessary to increase the maximum space allowed.)

Appendix A. BFX Error Messages

BFX generates a variety of messages during execution. Shown below is a complete list of messages with the suggested response for each. Also shown is the severity of the message (as compared with the MSGLEVEL parameter to determine if the message should be logged) and the modules that may issue the message.

Error messages reported by BFX are generally of the format:

```
BFXnnns message text
```

Where:

BFX Indicates that this is a BFX message.

nnn The message number. The BFX messages are listed in this order.

s The message severity. The following codes are used:

- **I** Informational
- **E** Error
- **S** Severe error
- **F** Fatal error

message text The message text.

The following are the messages issued by BFX.

BFX001F JOB SUBMISSION FAILED.

Severity: 15 (Fatal Error)

Explanation: Transfer Initiate was unable to submit a job to the remote host. If SOE was specified, the BFX program will terminate.

User Response: The reason for job submission failure will be indicated in a previous message. Take the corrective action indicated by the previous message's description.

BFX004I BFXJS STARTED.

Severity: 4 (Detailed informational)

Explanation: The BFXJS program has been started and is ready to offer Job Submission services.

User Response: None.

BFX006S "xxxxxxxx" NOT RECOGNIZED IN CONTROL STATEMENT.

Severity: 12 (Severe Error)

Explanation: An input statement to the BFX program contains a string that is not a recognized parameter. BFX will not transfer any files after encountering this error, but will continue to read the input file.

User Response: Correct the syntax error and resubmit the job.

BFX007W "xxxxxxxx" IS ONLY VALID FOR BFXTI JOBS – IGNORED.

Severity: 9 (Recoverable error)

Explanation: A parameter that is not applicable to the BFXTI program was encountered. The statement is ignored.

User Response: Although processing will continue, the probable cause is an operations or setup error. Verify that the remainder of the BFX run proceeded as intended.

BFX008W "xxxxxxxx" IS NOT VALID FOR A SUBMIT STATEMENT – IGNORED.

Severity: 9 (Recoverable Error)

Explanation: A parameter (such as BLOCK =) that is only used for file transfer was provided as an operand to the SUBMIT statement. The parameter is ignored and processing continues.

User Response: Although processing will continue, the probable cause is an operations or setup error. Verify that the remainder of the BFX run proceeded as intended.

BFX011F ID= BFX IDENTIFIER OMITTED.

Severity: 15 (Fatal Error)

Explanation: The ID parameter which uniquely identifies the BFX job on the initiating machine was not supplied. There is no default for this parameter.

User Response: Supply the ID parameter and rerun the job.

BFX014F ERRORS PREVIOUSLY FOUND. EXECUTION OF TRANSFER BYPASSED.

Severity: 13 (Severe Error)

Explanation: Errors were encountered parsing preceding input statements. The syntax of the input statements for following transfers will be checked, but the transfers will not occur.

User Response: Correct the errors indicated by the preceding error messages and resubmit the job.

BFX015I BFXTI STARTED.

Severity: 4 (Detailed informational)

Explanation: This message indicates that BFXTI has started and will begin to accept commands.

User Response: None.

BFX016I BFXTR STARTED.

Severity: 4 (Detailed informational)

Explanation: This message indicates that BFXTR has started and will begin to accept commands.

User Response: None.

BFX020F NRBSTAT = *ssss*, NRBIND = *iii* NETEX COMMUNICATIONS SUBSYSTEM IS NOT RUNNING.

Severity: 15 (Fatal error)

Explanation: When the BFX program attempted to establish communications, it found that the NETEX subsystem was not currently running on the local host. “ssss” is the four digit status code returned by NETEX; “iii” is the data or event indication type. Processing is terminated as data transfer is not possible without NETEX.

User Response: Consult with operations to determine whether NETEX should have been active. Resubmit the job when NETEX is active.

BFX021F NRBSTAT = *ssss*, NRBIND = *iii* NETEX COMMUNICATIONS SUBSYSTEM IS BEING SHUT DOWN.

Severity: 15 (Fatal error)

Explanation: During the connection process, or in the middle of a job or file transfer, the BFX program received an indication that NETEX is abruptly terminating. This can be caused by operator cancellation of NETEX or by internal NETEX software problems. “ssss” is the four digit status code returned by NETEX; “iii” is the data or event indication type. Processing is terminated, as no further data transfer will be possible until NETEX is restarted.

User Response: Consult with operations to determine the cause of the NETEX shutdown. Resubmit the job when NETEX is once again active. File cleanup procedures may be needed if a file transfer was in progress at the time of the failure.

BFX022F NRBSTAT = *ssss*, NRBIND = *iii* NETEX SYSTEMWIDE CAPACITY EXCEEDED.

Severity: 15 (Fatal error)

Explanation: During the process of establishing communications, NETEX returned an indication that it cannot handle a new connection because a limiting number of NETEX connections are already in use. “ssss” is the four digit status code returned by NETEX; “iii” is the data or event indication type. Processing is terminated, as it is uncertain when the condition will clear up.

User Response: Inform operations or the NETEX system programmer of the problem. If the problem occurs frequently, NETEX will have to be given more resources to handle extra connections.

BFX023F NRBSTAT = *ssss*, NRBIND = *iii* REMOTE BFX PROGRAM DID NOT START.

Severity: 15 (Fatal error)

Explanation: The corresponding BFX program was not present when required. If a BFXTI program issued this message, then it waited for the TIMEOUT= interval without being connected to by the BFXTR program. If a BFXTR program issued the message, then the originating BFXTI program is no longer present to be connected to. “ssss” is the four digit status code returned by NETEX; “iii” is the data or event indication type.

User Response: This is the error that will commonly occur if errors are made in the BFX setup. The most frequent causes of this error are:

- Job Control Language errors in the BFXTR job prevented successful execution of the BFXTR program.
- The TIMEOUT= value of the BFXTI job did not allow sufficient time for the BFXTR job to progress through the execution queue and connect to the originating program.
- The ID= fields of the two jobs did not agree with one another.

BFX024F NRBSTAT = *ssss*, NRBIND = *iii* REMOTE HOST CEASED COMMUNICATING.

Severity: 15 (Fatal error)

Explanation: During the transfer of a file or a job, the BFX program received an indication from NETEX that all communications with the other host have ceased. This is generally caused by a system crash on the remote host, abrupt failure or operator cancellation of NETEX on the remote host, or a hardware failure in the physical connection between the two hosts. “ssss” is the four digit status code returned by NETEX; “iii” is the data or event indication type. Processing is terminated, as no further data transfer is possible.

User Response: Consult with operations to determine the cause of the failure. Resubmit the job when the connection is once again active. File cleanup procedures may be needed if a file transfer was in progress at the time of the failure.

BFX025F NRBSTAT = *ssss*, NRBIND = *iii* REMOTE BFX ABORTED EXECUTION.

Severity: 15 (Fatal error)

Explanation: During the transfer of a file or a job, the BFX program received an indication from NETEX that the BFX program on the remote host terminated abnormally. “ssss” is the four digit status code returned by NETEX; “iii” is the data or event indication type.

User Response: Examine the output from the job on the remote host to determine the cause of failure. Correct the error and resubmit the job.

BFX026F NRBSTAT = *ssss*, NRBIND = *iii* REMOTE HOST NETEX NOT PRESENT.

Severity: 15 (Fatal Error)

Explanation: When an attempt was made to connect to the remote BFX program, the NetEx subsystem on the local machine reported that no NetEx subsystem was present on the remote host. “ssss” is the four digit status code returned by NETEX; “iii” is the data or event indication type. Processing is terminated, as data transfer is not possible without NetEx at that time.

User Response: Consult with operations to determine whether NetEx should have been present on the remote host. Resubmit the job when NetEx is available on both hosts.

BFX027F SPECIFIED HOST IS NOT ON THE NETWORK.

Severity: 15 (Fatal Error)

Explanation: When BFXTI was attempting to connect to BFXJS, or when BFXTR was attempting to connect back to the initiating BFXTI, the local NetEx subsystem returned an indication that the host name specified is not on the network specified in the Network Configuration Table (NCT). Processing is terminated, as data transfer is not possible.

User Response: The probable cause of this error is an erroneous HOST parameter. A second possibility is that the installation has changed the host names used by NetEx. Correct the error and resubmit the job.

BFX028F NRBSTAT = *ssss*, NRBIND = *iii* ACCESS TO SPECIFIED HOST DENIED.

Severity: 15 (Fatal Error)

Explanation: When BFXTI was attempting to connect to BFXJS, or when BFXTR was attempting to connect back to the initiating BFXTI, NetEx informed the program that access to the specified host has been denied by the local computer operator. “ssss” is the four digit status code returned by NETEX; “iii” is the data or event indication type. Processing is terminated, as communications between the two hosts cannot take place.

User Response: Computer operations is using a feature of NetEx that can temporarily prohibit access to a host that is undergoing maintenance, performing classified or confidential work, and so on. Consult with op-

erations to determine when communications with the remote host will once again be permitted. Resubmit the job at that time.

BFX029F NRBSTAT = *ssss*, NRBIND = *iii* ACCESS TO LOCAL NetEx DENIED.

Severity: 15 (Fatal Error)

Explanation: When the BFX program was attempting to establish communications, NetEx informed the program that access to the local host has been denied by the local computer operator. “ssss” is the four digit status code returned by NETEX; “iii” is the data or event indication type. Processing is terminated, as communications cannot take place.

User Response: Computer operations is using a feature of NetEx that can temporarily prohibit access to a host that is undergoing maintenance, performing classified or confidential work, and so on. Consult with operations to determine when communications with the remote local will once again be permitted. Resubmit the job at that time.

BFX030S NRBSTAT = ssss, NRBIND = iii, NetEx ERROR.

Severity: 12 (Severe Error)

Explanation: NetEx has reported an error to the BFX program that is not an intercepted condition. “ssss” is the four-digit status code returned by NetEx; “iii” is the data or event indication type. Processing is terminated, as the actual severity of the error is not known by the BFX program.

User Response: Refer to NetEx documentation to determine the cause of the error. Frequently this error will be caused by earlier, more comprehensible errors. If other BFX error messages precede this one, take the corrective action suggested by those messages.

BFX031S NRBSTAT = ssss, NRBIND = iii, SPECIFIED ID IS BUSY.

Severity: 12 (Severe Error)

Explanation: When BFXTI was attempting to connect to BFXJS, or when BFXTR was attempting to connect back to the initiating BFXTI, NetEx informed the program that the OFFERed application was currently in use by some other network application. The connection attempt was retried a number of times (until the DELAYTIME time period elapsed), but the application remained in use. “ssss” is the four-digit status code returned by NetEx; “iii” is the data or event indication type.

User Response: If the remote application was not expected to be busy, consult with operations. Otherwise, it may be necessary to specify a higher value for DELAYTIME and resubmit the job.

BFX032S NRBSTAT = ssss, NRBIND = iii, SPECIFIED ID NOT OFFERED ON SPECIFIED HOST.

Severity: 12 (Severe Error)

Explanation: When BFXTI was attempting to connect to BFXJS, or when BFXTR was attempting to connect back to the initiating BFXTI, NetEx informed the program that the OFFERed application was not currently available. The connection attempt was retried a number of times (until the DELAYNOFR time period elapsed), but the connection was never completed. Either BFXJS (first case above) or the initiating BFXTI (second case) failed before OFFERing itself, or response times on the remote machine are very slow. “ssss” is the four-digit status code returned by NetEx; “iii” is the data or event indication type.

User Response: Examine the output from the remote job to determine whether the job failed before OFFERing itself. If so, correct the error that caused the failure and resubmit the job. If this situation is caused by slow response times on the remote host, it may be necessary to specify a higher value for DELAYNOFR and resubmit the job.

BFX034S NRBSTAT = ssss, NRBIND = iii, READ OR OFFER TIMEOUT.

Severity: 12 (Severe Error)

Explanation: The timeout specified on an SREAD or SOFFR request has expired before the request was satisfied. For SOFFR, the probable cause is that the remote job did not start in time. “ssss” is the four-digit status code returned by NetEx; “iii” is the data or event indication type.

User Response: If nothing unusual is reported on the other side of the transfer, try setting TIMEOUT and/or TIMEOFFER to higher values.

BFX040F NRBSTAT = ssss, NRBIND = iii, NetEx COMMUNICATIONS SUBSYSTEM TERMINATED.

Severity: 15 (Fatal Error)

Explanation: During the connection process or in the middle of a job or file transfer, the BFX program received an indication that NetEx is abruptly terminating. This can be caused by operator cancellation of NetEx or by internal NetEx software problems. “ssss” is the four-digit status code returned by NetEx; “iii” is the data or event indication type. Processing is terminated, as no further data transfer will be possible until NetEx is restarted.

User Response: Consult with operations to determine the cause of the NetEx shutdown. Resubmit the job when NetEx is once again active. File cleanup procedures may be needed if a file transfer was in progress at the time of the failure.

BFX042F NRBSTAT = ssss, NRBIND = iii, BFX PROGRAM TIMED OUT TO NETEX.

Severity: 15 (Fatal Error)

Explanation: The BFX program suspended execution for a sufficiently long period of time that NETEX terminated the connection between the two BFX programs. The current transfer is aborted, but the remaining transfers will be attempted. “ssss” is the four-digit status code returned by NetEx; “iii” is the data or event indication type.

User Response: This is generally because of difficulties in system tuning, or exceptionally long delays in such activities as tape mounting. If the problem was not caused by operational errors, the NETEX system programmer may have to raise the NETEX READTO parameter to compensate for the long delay.

BFX043F BFX PROTOCOL ERROR – PREMATURE DISCONNECT.

Severity: 15 (Fatal Error) BFXSND in BFXTI and BFXTR.

Explanation: The remote BFX program terminated the connection at a time when termination was not anticipated by the local BFX program.

User Response: This is an internal BFX error. It should be brought to the attention of installation BFX support personnel.

BFX044F NRBSTAT = ssss, NRBIND = iii, REMOTE BFX PROGRAM TIME OUT.

Severity: 15 (Fatal Error)

Explanation: The remote BFX program suspended execution for a sufficiently long period of time that NETEX terminated the connection between the two BFX programs. The current transfer is aborted, but the remaining transfers will be attempted. “ssss” is the four-digit status code returned by NetEx; “iii” is the data or event indication type.

User Response: This is generally because of difficulties in system tuning, or exceptionally long delays in such activities as tape mounting. If the problem was not caused by operational errors, the NETEX system programmer may have to raise the NETEX READTO parameter to compensate for the long delay.

BFX045F BFX PROTOCOL ERROR – PREMATURE END MESSAGE.

Severity: 15 (Fatal Error) BFXRCV in BFXJS, BFXTI, and BFXTR.

Explanation: The remote BFX program sent an End-of-File message before the End-of-File record was received.

User Response: This is generally caused by user-written block and/or record modules. Re-code the user module to send the last record of the file with an EOF record level and then send the End-of-File message.

BFX046F BFX PROTOCOL ERROR – DATA AFTER EOF.

Severity: 15 (Fatal Error)

Explanation: The remote BFX program sent data after sending a record with an EOF record level.

User Response: This is generally caused by user-written block and/or record modules. Re-code the user module to send only the last record of the file with an EOF record level.

BFX047I JOB ssss SUBMITTED.

Severity: 7 (Informational)

Explanation: The job file sent to BFXJS was submitted to the system batch queue with job identifier (SNUMB) = ssss.

User Response: None.

BFX048S JOB SUBMISSION FAILED. RC = cc

Severity: 12 (Severe Error)

Explanation: The job file submission failed. The error is described by the system status code “cc”.

User Response: Correct the error indicated by the status code and resubmit the job.

BFX070W aaaa = bbbb; PARAMETER VALUE MISSING, INVALID, OR OUT OF RANGE.

Severity: 12 (Warning)

Explanation: The parameter “aaaa” is either missing a value or has an incorrect value specified.

User Response: Correct the error and resubmit the job.

BFX080I FILE ffffffff RECEIVED.

Severity: 6 (Informational)

Explanation: The file ffffffff was received. This message is generated if the receiving record module does not return a message on EOF.

User Response: None.

BFX081F RECORD MODULE RETURNED A DATA RECORD DURING INITIALIZATION.

Severity: 15 (Fatal Error)

Explanation: A record module returned a data record during initialization. This is generally caused by incorrectly written user record modules.

User Response: Recode the user module so that data records are not returned on open calls.

BFX082I FILE ffffffff SENT.

Severity: 6 (Informational)

Explanation: The file ffffffff was sent. This message is generated if the sending record module does not return a message on EOF.

User Response: None.

BFX083S FILE ffffffff ABORT PROCESSED.

Severity: 12 (Severe Error)

Explanation: A record module has requested an abort.

User Response: Correct the condition that caused the user module to return the abort code and resubmit the job. It is good form for user modules to supply a message under these circumstances.

BFX084S PROTOCOL ERROR – DATA RECEIVED WHEN SENDING.

Severity: 13 (Severe Error)

Explanation: Unexpected data was received when sending.

User Response: This is generally caused by user-written block and/or record modules. Re-code the user module to correct the problem.

BFX085F MESSAGE IN DATA BLOCK.

Severity: 15 (Fatal Error)

Explanation: A message record was found inside a data block. Messages must appear in their own blocks, one to a block.

User Response: This is generally caused by user-written block module. Correct the block module and resubmit the job.

BFX088S OFFER OF hhhhhhhh FAILED.

Severity: 12 (Severe Error)

Explanation: Typically the remote BFX has not issued its connect or has timed out.

User Response: Verify the remote application status and try again.

BFX089S CONNECT TO hhhhhhhh FAILED.

Severity: 12 (Severe Error)

Explanation: Typically the remote BFX application has not issued its offer or its offer has timed out.

User Response: Verify the remote application status and try again

BFX101I FILE ffffffff DONE; nnnn RECORDS SENT.

Severity: 6 (Informational)

Explanation: The NetEx Software standard Sending Record Module has detected normal end of file on the input file specified by “ffffffff”. The total number of logical records sent for this particular file is “nnnn”. At the time the message was issued, the last record of the file will already have been sent to the receiving BFX.

User Response: None.

BFX102S FILE ffffffff PERMANENT I/O ERROR – RC = rrrr.

Severity: 12 (Severe Error) BFXRRM in BFXTI, BFXTR, and BFXJS.

Explanation: The access method reported a permanent I/O error reading or writing a file during transfer of the job or transfer of the file. The file is indicated by "ffffffff". Transfer of this file is aborted. If batch transfer of files is being performed, BFX will attempt to transfer the rest of the specified files.

User Response: Determine the cause of the I/O error. If the error can be corrected, rerun the BFX jobs.

BFX103S CANNOT FINE RECORD MODULE rrrrrrrr.

Severity: 15 (Fatal Error)

Explanation: User specified a record module to be used by BFX but it could not be found.

User Response: Re-build BFX with the desired module(s) included or remove the RMOD parameter.

BFX104F STOP ON ERROR SET, ERRORS ENCOUNTERED.

Severity: 15 (Fatal Error)

Explanation: A previous error was detected and the SOE parameter was declared to stop on errors.

User Response: Locate the previous error and correct it.

BFX107S CANNOT FIND JOB MODULE jjjjjjj.

Severity: 12 (Severe Error)

Explanation: The Job Submission Module specified by the JMOD parameter was not compiled into the current copy of the BFX program. The current transfer is aborted, but the remaining transfers will be attempted.

User Response: This error will be seen either because the BFXUIM module was not updated to include the job submission module requested, or because of use of the incorrect module name.

BFX111S RECORD MODULE INITIALIZATION FAILED.

Severity: 12 (Severe Error) BFXRBM in BFXJS, BFXTI, and BFXTR.

Explanation: A user-written Record Module returned an Abort code (Buflev = 16) when called for initialization. The module did not supply a message with the abort code, so this default message is printed. The current transfer is aborted, but the remaining transfers will be attempted.

User Response: Correct the condition that caused the user module to return the Abort code and resubmit the job. It is good form for user modules to supply a message under these circumstances.

BFX113S SENDING RECORD MODULE ABORTED TRANSFER.

Severity: 12 (Severe Error)

Explanation: A user-written Sending Record Module returned an Abort code (Buflev = 16) during the transfer of a file. The user module did not supply a message with the abort code, so this default message is printed. The current transfer is aborted, but the remaining transfers will be attempted.

User Response: Correct the condition that caused the user module to return the abort code and resubmit the job. It is good form for user modules to supply a message under these circumstances.

BFX114S RECEIVING RECORD MODULE ABORTED TRANSFER.

Severity: 12 (Severe Error)

Explanation: A user-written Receiving Record Module returned an Abort code (Buflev = 16) during the transfer of a file. The user module did not supply a message with the abort code, so this default message is printed. The current transfer is aborted, but the remaining transfers will be attempted.

User Response: Correct the condition that caused the user module to return the Abort code and resubmit the job. It is good form for user modules to supply a message under these circumstances.

BFX115S SENDING BLOCK MODULE ABORTED TRANSFER.

Severity: 12 (Severe Error)

Explanation: A user-written Sending Block Module returned an Abort code (Buflev = 16) during the transfer of a file. The user module did not supply a message with the abort code, so this default message is printed. The current transfer is aborted, but the remaining transfers will be attempted.

User Response: Correct the condition that caused the user module to return the Abort code and resubmit the job. It is good form for user modules to supply a message under these circumstances.

BFX116S RECEIVING BLOCK MODULE ABORTED TRANSFER.

Severity: 12 (Severe Error)

Explanation: A user-written Receiving Block Module returned an Abort code (Buflev = 16) during the transfer of a file. The user module did not supply a message with the abort code, so this default message is printed. The current transfer is aborted, but the remaining transfers will be attempted.

User Response: Correct the condition that caused the user module to return the Abort code and resubmit the job. It is good form for user modules to supply a message under these circumstances.

BFX121S MAXIMUM RECORD LENGTH EXCEEDS NEGOTIATED SIZE.

Severity: 12 (Severe Error)

Explanation: When the two BFX programs established a connection, the negotiated block size as determined by the user-specified parameters was insufficient to hold the largest logical record in the file to be sent. The current transfer is aborted, but the remaining transfer will be attempted.

User Response: Adjust the BLOCK parameter in one of the two BFX programs so it is sufficient to transfer the file. Note that a header of 6 bytes (bit mode) or 8 bytes (character mode) is prefixed to each record transferred.

BFX123F FILE TRANSFER PROTOCOL SEQUENCE ERROR.

Severity: 15 (Fatal error)

Explanation: The file transfer information sent to the receiving BFX was found to be incorrect. A record numbering check indicated that records are missing, duplicated, or out of sequence. This may be because of an internal BFX or NETEX error, or to a user-written Block Module that is incorrectly sending data to the standard NetEx Software Receiving Block Module. The current transfer is aborted, but the remaining transfer will be attempted.

User Response: If the error was caused by a user-written Block Module, correct the coding error that caused incorrect data to be sent. If only NetEx Software BFX code was used, bring the error to the immediate attention of NetEx Software's Customer Support.

BFX124S MODE= PARAMETERS NOT CONSISTENT FOR BOTH BFX JOBS.

Severity: 12 (Severe error) BFXSCN in BFXTI and BFXTR.

Explanation: In a BFX program pair, one side had MODE = BIT specified and the other had MODE = CHAR, MODE = BCD, or MODE = ASCII. The current transfer is aborted, but the remaining transfers will be attempted.

User Response: Correct the erroneous specification and transfer the files that were not sent.

BFX125S MAXIMUM RECORD LENGTH EXCEEDS BUFFER SIZE.

Severity: 12 (Severe error)

Explanation: The length of the longest record in the file to be sent (or the physical block size of a sequential-only device) exceeds the length of the BFX buffers. The size of the buffers is determined by the default NETEX block size or a BFX BLOCK command. The current transfer is aborted, but the remaining transfers will be attempted.

User Response: Increase the BFX block size using the BLOCK command.

BFX201I FILE ffffffff DONE; nnnnnnnn RECORDS RECEIVED.

Severity: 6 (Informational)

Explanation: This message is issued when the receiving BFX processes the last record of the file. When issued, it indicates that the last record was received and that the output file was successfully closed. “ffffffff” is the logical name of the file used for output; “nnnnnnnn” is the number of records that were written to the output file.

User Response: None.

BFX202W FILE ffffffff DONE; nnnnnnnn RECORDS RECEIVED, yyyy TRUNCATED.

Severity: 9 (Warning)

Explanation: The file was received as described for message BFX201I, but “yyyy” records were truncated.

User Response: If the file content is incorrect, resubmit the job with the correct RMAXL parameter.

BFX203E FILE ffffffff TRANSFER ABORTED; nnnnnnnn RECORDS RECEIVED.

Severity: 10 (Error)

Explanation: This message is issued by the receiving BFX when the file transfer process is aborted either because of the loss of NETEX communication or because of some other error detected by the BFX program. “ffffffff” is the logical name of the file used for output; “nnnnnnnn” is the number of records that were written to the output file before the abort caused the transfer to stop. The current transfer is aborted, but the remaining transfers will be attempted. The original error will be reported by other BFX messages.

User Response: Correct the error that caused the abort. Transfer the file again.

BFX204E FILE ffffffff TRANSFER ABORTED; nnnnnnnn RECORDS RCVD, yyyy TRUNCATED.

Severity: 11 (Error)

Explanation: In addition to the abort condition as described for message BFX203E, “yyyy” records that were received were truncated.

User Response: In addition to fixing the cause of the abort, it may be necessary to increase the RMAXL parameter.

BFX205W FILE ffffffff DONE; nnnn RECORDS SENT, yyyy TRUNCATED.

Severity: 9 (Error)

Explanation: The file was successfully sent, but “yyyy” records were truncated,

User Response: Check the data file on the receiving side, and if all desired data did not get sent, resubmit the job with a larger RMAXL parameter.

BFX206E FILE ffffffff TRANSFER ABORTED; nnnnnnnn RECORDS SENT.

Severity: 10 (Error)

Explanation: This message is issued by the sending BFX when the file transfer process is aborted either because of the loss of NETEX communication or because of some other error detected by the BFX program. “ffffffff” is the logical name of the file used for input; “nnnnnnnn” is the number of records that were read from the input file before the abort caused the transfer to stop. The current transfer is aborted, but the remaining transfers will be attempted. The original error will be reported by other BFX messages.

User Response: Correct the error that caused the abort. Transfer the file again.

BFX210S CANNOT OPEN INPUT FILE ffffffff. RC = cccccccc.

Severity: 12 (Severe error).

Explanation: The sending BFX program was unable to open the input file whose logical name is specified by “ffffffff”. The return code “cccccccc” is in decimal. The return code may be a return code from either UFAS or FMS. Detailed descriptions of these codes may be found in the Bull UFAS and FMS manuals.

A value of 1 for the return code indicates that the data mode is not supported. A value of 6 indicates that the record size is larger than BFX is prepared to handle. A value of 9999 indicates an error detected by the ATTACH routine that is neither a UFAS nor an FMS error. This can be caused by using the wrong allocation mode for a transfer. The execution report often contains messages produced by UFAS or FMS describing the error in more detail.

User Response: Correct the reason for the open failure. Transfer the file again.

BFX211S CANNOT OPEN OUTPUT FILE ffffffff. RC = cccccccc.

Severity: 12 (Severe error).

Explanation: The sending BFX program was unable to open the output file whose logical name is specified by “ffffffff”. The return code “cccccccc” is in decimal. The return code may be a return code from either UFAS or FMS. Detailed descriptions of these codes may be found in the Bull UFAS and FMS manuals.

A value of 1 for the return code indicates that the data mode is not supported. A value of 6 indicates that the record size is larger than BFX is prepared to handle. A value of 9999 indicates an error detected by the ATTACH routine that is neither a UFAS nor an FMS error. This can be caused by using the wrong allocation mode for a transfer. The execution report often contains messages produced by UFAS or FMS describing the error in more detail.

User Response: Correct the reason for the open failure. Transfer the file again.

BFX212S FILE ffffffff PERMANENT I/O ERROR. RC= cccccccc.

Severity: 12 (Severe error) BFXRRM in BFXJS, BFXTI, and BFXTR.

Explanation: During the process of reading or writing the file whose logical name is "ffffffff", a permanent I/O error occurred. The return code "cccccccc" is in decimal. The return code is from UFAS for character transfers. The execution report will have the UFAS message associated with the error.

For bit transfers, the return code is usually the value of bits 0-11 of the status return word returned when trying to read or write a file. A return code of 5 indicates the file size could not be expanded. The execution report will generally have any additional information about the error.

User Response: Determine the cause of the I/O error. If the error can be corrected, do so and transfer the file again.

BFX220I SENDING FILE ffffffff AT hh mm ss.sss.

Severity: 4 (Informational).

Explanation: The sending BFX has successfully opened the input file and is ready to begin transfer of data. Transmission will begin as soon as this message is issued. "ffffffff" is the logical name of the input file, followed by the sending local time of day in standard format.

User Response: None.

BFX221I RECEIVING FILE ffffffff AT hh mm ss.sss

Severity: 4 (Informational).

Explanation: The receiving BFX has successfully opened the output file and is ready to receive file data. "ffffffff" is the logical name of the output file, followed by the receivers local time of day in standard format.

User Response: None.

BFX301I OFFERING ssssssss; BLOCK IN SIZE bbbb.

Severity: 2 (Diagnostic).

Explanation: The BFX program has issued a NETEX SOFFER to wait for the responding BFX program to connect to it. The name offered is "ssssssss", which is the JID or ID parameter specified in an input statement. The block size that the offering program would like use is "bbbb".

User Response: None.

BFX302I CONNECTING TO ssssssss ON HOST hhhhhhhh; BLOCK IN SIZE nnnn.

Severity: 2 (Diagnostic).

Explanation: When BFXTR is connecting back to its starting BFXTI, it has issued a NETEX SCONNECT to establish communications. "ssssssss" is the name to connect to as specified in the ID= or JID= user parameters; "hhhhhhhh" is the host name specified in the TO= or FROM= parameters. The direction of file transfer will cause this program to receive the file; the Block size that this program is prepared to receive is "nnnn".

User Response: None.

BFX304I CONNECT COMPLETE.

Severity: 2 (Diagnostic).

Explanation: A previously issued NETEX SCONNECT has completed successfully.

User Response: None.

BFX307I OFFER COMPLETED FROM HOST hhhhhhhh

Severity: 2 (Diagnostic).

Explanation: A previous NETEX SOFFER request has completed successfully. ‘hhhhhhhh’ is the host name.

User Response: None.

BFX308I CONFIRM ISSUED.

Severity: 2 (Diagnostic).

Explanation: A NETEX SCONFIRM is being issued in response to a previously completed SOFFER.

User Response: None.

BFX309I CONFIRM COMPLETE.

Severity: 2 (Diagnostic).

Explanation: A previously issued NETEX SCONFIRM has completed successfully.

User Response: None.

BFX310I CONNECT CONFIRM READ ISSUED.

Severity: 2 (Diagnostic).

Explanation: Following a successful SCONNECT request, the BFX program has issued an SREAD to obtain the SCONFIRM response from the other program.

User Response: None.

BFX311I CONNECT CONFIRM COMPLETE

Severity: 2 (Diagnostic).

Explanation: The SREAD issued to accept an SCONFIRM message from the remote BFX program has completed successfully. The NETEX session establishment process is now complete.

User Response: None.

BFX312I BLOCK SIZE bbbbb, DATAMODE dddd, LCM lll.

Severity: 2 (Diagnostic).

Explanation: This message is issued by the BFX program when the session negotiation process is complete. “bbbbb” is the NETEX block size that will be used during file transfer. “dddd” gives the NETEX DATAMODE to be used based on the requirements of the two BFX programs. “lll” is the Least Common Multiplier size negotiated.

User Response: None.

BFX402S VALUE MUST BE SPECIFIED FOR THIS PARAMETER – NO DEFAULT.

Severity: 12 (Severe error).

Explanation: No value was specified in an input statement to be assigned to a parameter that does not have a default value (such as ID). BFX will not transfer any files after encountering this error, but will continue to read the input file.

User Response: Supply a value for the parameter and resubmit the job.

BFX403S MODE MUST BE BIT OR A VALID CHARACTER SET.

Severity: 12 (Severe error).

Explanation: A string (or abbreviation) other than “BIT”, “BCD”, “ASCII”, or “CHARACTER” was specified in a MODE= input statement. BFX will not transfer any files after encountering this error, but will continue to read the input file.

User Response: Correct the incorrect string and resubmit the job.

BFX407E PARAMETER VALUE TOO LONG; TRUNCATED. MAXIMUM:

Severity: 9 (Error).

Explanation: A string value specified in an input statement to be assigned to a parameter was longer than the parameter field itself. The string is truncated to fit in the field. The maximum length of the field in question is printed following the message. Processing will continue normally.

User Response: Unexpected results may occur because of this truncation. If so, supply a valid length string and resubmit the job.

BFX408I ALL FILE TRANSFERS HAVE BEEN PROCESSED.

Severity: 4 (Informational)

Explanation: BFX actions have completed.

User Response: None.

BFX409S NUMERIC VALUE REQUIRED FOR xxxx, yyyy GIVEN.

Severity: 12 (Severe error).

Explanation: An input token requiring a numeric parameter was not given one.

User Response: Fix the input stream and re-submit.

BFX410S VALUE xxxx OUT OF RANGE ON yyyy COMMAND.

Severity: 12 (Severe error).

Explanation: The parameter value xxxx on command yyyy exceeded the allowed range.

User Response: Select a valid value and reissue the command.

BFX901I H291 b.r dddd, SNUMB ssss, TYPE ttttt.

Severity: 15 (Informational – always issued).

Explanation: BFX sign-on line. “b.r” is the release identifier (for example, 2.2), “dddd” is the year and mod of release, “sssss” is the job SNUMB identifier, “ttttt” is the type (for example, BFXTI).

User Response: None.

BFX902W DATA MODE NOT IMPLEMENTED: xxx.

Severity: 12 (Severe Error)

Explanation: The MODE value specified is not supported in this BFX implementation. BFX will not transfer any files after encountering this error, but will continue to read the input file. “xxx” is the MODE requested.

User Response: Supply a valid supported MODE parameter value and resubmit the job.

BFX903F FORMAT MUST BE GFRC, UFAS, ANSI, IBM, or HB.

Severity: 12 (Severe error).

Explanation: The FORMAT value specified is not a valid file format for this implementation of BFX. BFX will not transfer any files after encountering this error, but will continue to read the input file.

User Response: Supply a valid supported FORMAT parameter value and resubmit the job.

BFX904W FIRST RECORD OF SPAWNED JOB MUST BE DUMMY OR SNUMB.

Severity: 12 (Severe error)

Explanation: The first JCL statement of a job file must be $ DUMMY or $ SNUMB, and for $ SNUMB, columns 16-20 must contain a valid job identifier subject to GCOS restrictions.

User Response: Correct the Job File and re-submit.

Appendix B. Abort Processing

The BFX user can control the actions of the 'process if a fatal error occurs during the processing of file transfers. This is accomplished with the Process Switch Word (PSW).

If the user wants to check the status of the transfer with JCL statements instead of having to check the execution report for a normal or abnormal termination, switch 19 should be set on before the activity, and switch 18 should be tested at the conclusion of the activity.

For example:

```
$     SET     19
$     PROGRAM BFXTI
      .
      .       (OTHER JCL STATEMENTS & BFX COMMANDS...)
      .
$     IF      18,ITABTD
      .
      .       (PROCEDURES AFTER SUCCESSFUL TRANSFER)
      .
$ ITABTD:     (COME HERE IF TRANSFER ABORTED)
      .
      .
      .
```

(See the appropriate Bull JCL reference manual for further information on these statements.)

During BFX initiation, switch 18 will be set ON, and the state of switches 19-35 will be stored internally and set OFF. If no transfer errors occur, and the activity terminates normally, switch 18 will be set OFF, and switches 19-35 will be reset to their initial state.

If a fatal error condition occurs, and switch 19 was ON at the start of the activity, switch 18 will be set ON, and switches 19-26 will be set to the HEX equivalent of the abort error code. For example, an abort code ‘8E’ would result in switches 19, 23, 24, and 25 being set ON, and 20, 21, 22, and 26 being set OFF. (See the table of error codes in Table 8 on page 70.)

If the process is aborted from an external source, (for example, operator abort or GCOS detected error), or because of error conditions detected within various procedures such as UFAS or various ‘B’ library routines, wrap-up processing will always set switches 18-26 ON. If for some reason wrap-up is not paid, or it is unsuccessful, switch 18 will still reflect the abnormal end of activity since it was set ON during initiation.

It is, of course, possible to be aborted within the process setup procedures (.SETU.) before BFX initialization can take place. One case of this might be the inability to initialize the PASCAL STACK or HEAP space because of core limits. One method of still providing the switch testing capability would be to set both switches 18 and 19 on with a $ SET JCL statement before the BFX activity.

Table 3. Abort Codes		
Origin	**Code**	**Description**
BFX (main)	21	Syntax
BFXUIM	22	Bad block or record module name.
BFXSCN/BFXSOF	34	SOFFR/SCONN retries exhausted
	3E	NETEX returned an error on SCONN/SOFFR
	4E	NETEX returned an error on SREAD/SCONF
BFXRCV/BFXSND	6D	Premature Disconnect during file transfer
TERMFILE (called by BFXRCV/BFXSND with error code set)	81	BMOD, error returned by file open routine.
	82	End message received, but not end of file.
	84	Data received while sending.
	88	Remote side sent level 4 error message.
	8E	NETEX returned an error status in NRBSTAT.
TERMCONN (called by BFXRCV/BFXSND with error code set)	91	BMOD/RMOD aborted file transfer.
	92	Data received after end of file (BFXRCV)
	98	Data received while sending (BFXSND). Remote side sent level 4 error message.
	9D	Remote side disconnected early. (BFXSND)
	9E	NETEX returned an error status in NRBSTAT.
Other	E1	Record Sequence Error.
	E2	Message in Data Block
	TM	Unable to establish TCP connection with TNP