BSCTRAN
Programmer's Guide

DC 900-1406E

Simpact, Inc.
9210 Sky Park Court
San Diego, CA 92123
March 1999

SIMPACT

Simpact, Inc.
9210 Sky Park Court
San Diego, CA 92123
(619) 565-1865

BSCTRAN Programmer's Guide

Printed in the United States of America

Contents

List of Figures

List of Tables

Preface

Purpose of Document

This document describes the operation and programming interface required to use Simpact's BSCTRAN product running on Simpact's Freeway communications server or embedded ICP.

Note

In this document, the term "Freeway" can mean either a Freeway server or an embedded ICP. For the embedded ICP, also refer to the user's guide for your ICP and operating system (for example, the *ICP2432 User's Guide for OpenVMS Alpha (DLITE Interface)*).

Intended Audience

This document should be read by programmers who wish to do any of the following:

- Move a file to or from a BSC wide-area network using the BSCTRAN software, the Freeway software, and the BSC communications software
- Write a file transfer application program to link with the BSCTRAN file transfer interface (Chapter 3) or record transfer interface (Chapter 4)

Required Equipment

The BSCTRAN product requires the following two major hardware components to operate:

- A Freeway communications server or an embedded ICP that runs the BSC communications software
- VAX/VMS or UNIX client computer that runs the following:
  - TCP/IP (for a Freeway server)
  - Data link interface (DLI)

Note

BSCTRAN currently supports VMS and UNIX only.

Organization of Document

Chapter 1 gives an overview of the BSCTRAN File Transfer program.

Chapter 2 describes the User Interface (BSCTRAN) and gives details on using each of the commands and their options. This chapter will interest all users of the BSC File Transfer program.

Chapter 3 and Chapter 4 describe the File Transfer Interface (BSCFTI) and the Record Transfer Interface (BSCRTI) and each of their procedures. These chapters will interest users who wish to develop their own specialized user-oriented or file-oriented modules to link with the lower-level interfaces of the BSC File Transfer program.

Chapter 5 describes the error handling performed by the BSCTRAN program.

Chapter 6 lists cautions for using the BSCTRAN program. Read this chapter before attempting to run the program for the first time.

Chapter 7 shows an example of a command procedure that demonstrates most of the commands and options of the BSCTRAN program.

Simpact References

The following documents provide useful supporting information, depending on the customer's particular hardware and software environments. Most documents are available on-line at Simpact's web site, www.simpact.com.

General Product Overviews

- *Freeway 1100 Technical Overview* 25-000-0419
- *Freeway 2000/4000/8800 Technical Overview* 25-000-0374
- *ICP2432 Technical Overview* 25-000-0420
- *ICP6000X Technical Overview* 25-000-0522

Hardware Support

- *Freeway 1100/1150 Hardware Installation Guide* DC 900-1370
- *Freeway 1200 Hardware Installation Guide* DC 900-1537
- *Freeway 1300 Hardware Installation Guide* DC 900-1539
- *Freeway 2000/4000 Hardware Installation Guide* DC 900-1331
- *Freeway 8800 Hardware Installation Guide* DC 900-1553
- *Freeway ICP6000R/ICP6000X Hardware Description* DC 900-1020
- *ICP6000(X)/ICP9000(X) Hardware Description and Theory of Operation* DC 900-0408
- *ICP2424 Hardware Description and Theory of Operation* DC 900-1328
- *ICP2432 Hardware Description and Theory of Operation* DC 900-1501
- *ICP2432 Hardware Installation Guide* DC 900-1502

Freeway Software Installation Support

- *Freeway Software Release Addendum: Client Platforms* DC 900-1555
- *Freeway User's Guide* DC 900-1333
- *Getting Started with Freeway 1100/1150* DC 900-1369
- *Getting Started with Freeway 1200* DC 900-1536
- *Getting Started with Freeway 1300* DC 900-1538
- *Getting Started with Freeway 2000/4000* DC 900-1330
- *Getting Started with Freeway 8800* DC 900-1552
- *Loopback Test Procedures* DC 900-1533

Embedded ICP Installation and Programming Support

- *ICP2432 User's Guide for Digital UNIX* DC 900-1513
- *ICP2432 User's Guide for OpenVMS Alpha* DC 900-1511
- *ICP2432 User's Guide for OpenVMS Alpha (DLITE Interface)* DC 900-1516
- *ICP2432 User's Guide for Windows NT* DC 900-1510
- *ICP2432 User's Guide for Windows NT (DLITE Interface)* DC 900-1514

Application Program Interface (API) Programming Support

- *Freeway Data Link Interface Reference Guide* DC 900-1385
- *Freeway Transport Subsystem Interface Reference Guide* DC 900-1386
- *QIO/SQIO API Reference Guide* DC 900-1355

Socket Interface Programming Support

- *Freeway Client-Server Interface Control Document* DC 900-1303

Toolkit Programming Support

- *Freeway Server-Resident Application and Server Toolkit Programmer's Guide* DC 900-1325
- *OS/Impact Programmer's Guide* DC 900-1030
- *Protocol Software Toolkit Programmer's Guide* DC 900-1338

Protocol Support

- *ADCCP NRM Programmer's Guide* DC 900-1317
- *Asynchronous Wire Service (AWS) Programmer's Guide* DC 900-1324
- *Addendum: Embedded ICP2432 AWS Programmer's Guide* DC 900-1557
- *AUTODIN Programmer's Guide* DC 908-1558
- *BSC Programmer's Guide* DC 900-1340
- *BSCDEMO User's Guide* DC 900-1349
- *BSCTRAN Programmer's Guide* DC 900-1406
- *DDCMP Programmer's Guide* DC 900-1343
- *FMP Programmer's Guide* DC 900-1339
- *Military/Government Protocols Programmer's Guide* DC 900-1602
- *SIO STD-1200A (Rev. 1) Programmer's Guide* DC 908-1359

- *SIO STD-1300 Programmer's Guide* DC 908-1559
- *X.25 Call Service API Guide* DC 900-1392
- *X.25/HDLC Configuration Guide* DC 900-1345
- *X.25 Low-Level Interface* DC 900-1307

Document Conventions

The term "Freeway" refers to any of the Freeway server models (for example, Freeway 1100/1150/1200/1300, Freeway 2000/4000, or Freeway 8800), or to the embedded ICP product (for example, the embedded ICP2432).

Physical "ports" on the ICPs are logically referred to as "links." However, since port and link numbers are usually identical (that is, port 0 is the same as link 0), this document uses the term "link."

Program code samples are written in the "C" programming language.

Revision History

The revision history of the *BSCTRAN Programmer's Guide*, Simpact document DC 900-1406E, is recorded below:

Revision	Release Date	Description
DC 900-1406A	October 1994	Original release
DC 900-1406B	January 1995	Modified file name conventions
DC 900-1406C	January 1996	Add UNIX support
DC 900-1406D	February 1997	BSCTRAN supports VMS and UNIX only. Add note on page 32 for `SET /START` command. Add temporary recompile caution for Solaris 2.5.
DC 900-1406E	March 1999	Modifications for embedded ICP and DLITE support. Add `SET /LOCKSTART` command (Section 2.4 on page 31).

Customer Support

If you are having trouble with any Simpact product, call us at 1-800-275-3889 Monday through Friday between 8 a.m. and 5 p.m. Pacific time.

You can also fax your questions to us at (619)560-2838 or (619)560-2837 any time. Please include a cover sheet addressed to "Customer Service."

We are always interested in suggestions for improving our products. You can use the report form in the back of this manual to send us your recommendations.

Chapter

1 BSCTRAN File Transfer Program Overview

Note

In this document, the term "Freeway" can mean either a Freeway server or an embedded ICP. For the embedded ICP, also refer to the user's guide for your ICP and operating system (for example, the *ICP2432 User's Guide for OpenVMS Alpha (DLITE Interface)*).

BSCTRAN is a client-resident application that interfaces with the BSC 2780/3780 software on Freeway. BSCTRAN enables a user to send and receive text and binary files using one or more of the BSC communications lines on the ICP board. Because each link on the ICP is used independently, different links can be used simultaneously by different users or processes.

The BSC File Transfer program (BSCTRAN) was developed for the following users:

- VMS users who want to connect their VAX computers with IBM 2780/3780 Remote Job Entry (RJE) stations for the purpose of sending and receiving files. The BSCTRAN program can also be used to send and receive data between two VMS sites.
- UNIX users who want to perform UNIX-to-UNIX file transfers

Note

BSCTRAN currently supports VMS and UNIX only.

1.1 Freeway Server versus Embedded ICP Version

The Freeway server version of BSCTRAN uses the Freeway data link interface (DLI) application program interface (API). Its execution requires both a DLI configuration file (`dlitrancfg`) and a transport subsystem interface (TSI) configuration file (`tsitrancfg`). When BSCTRAN is executed, binary versions of `dlitrancfg` and `tsitrancfg` are read and the information used to manage the session(s) with Freeway. The `dlicfg` program is used to compile the DLI configuration text into binary form. The `tsicfg` program is used to compile the TSI configuration text into binary form. The binary versions of the configuration files are called `dlitrancfg.bin` and `tsitrancfg.bin`, respectively. These compilations take place prior to executing BSCTRAN. The binary files reside in the same directory as the BSCTRAN executable image. Refer to the *Freeway Data Link Interface Reference Guide* for more information.

The embedded ICP interface to DLI is called DLITE. The embedded ICP version is similar to the Freeway server version except that all ICPs are embedded on the client processor, the DLI configuration file is named `dlitetrancfg`, and there is no TSI configuration file. The DLITE interface is described in the user's guide for your ICP and operating system (for example, the *ICP2432 User's Guide for OpenVMS Alpha (DLITE Interface))*.

1.2 BSCTRAN Modules

The BSCTRAN program contains five modules that represent different levels of interface. The modular nature of the interfaces allows the user to develop special applications linked with a subset of these modules to provide either a file-oriented or a record-oriented interface. All five modules are written in the C programming language and source code is provided so that the user can make modifications if needed. Two include files (`BSC.H` and `BSCTRAN.INC`) are provided for compilation. This document describes the top three BSC interfaces: BSCTRAN, BSCFTI, and BSCRTI.

The five modules are as follows:

1. User-interactive or Batch Program (BSCTRAN)
2. File Transfer Interface (BSCFTI)
3. Record Transfer Interface (BSCRTI)
4. Freeway Conversion Module (BSCFWY)
5. DLI Interface (ICPDLI)

The BSCTRAN program runs as an executable image and accepts commands from a terminal or a command file. BSCTRAN can be run in a batch or an interactive environment. The program accepts commands to send or receive files across a BSC communications line. BSCTRAN is linked with the other four modules that are subroutines that perform lower-level I/O functions.

The file transfer interface (BSCFTI) consists of a set of subroutines that provide file manipulation for BSCTRAN (for example, the VMS RMS handling). These routines can also be called by a user-written application (other than BSCTRAN).

The record level interface (BSCRTI) subroutines handle BSC record packing and unpacking. These routines can also be called by a user-written application (other than BSCTRAN).

The BSCFWY module provides command and subroutine conversion for the Freeway product. BSCFWY is used in place of BSCICP which originally provided an interface to a Simpact ICP3222 (Q-bus) board. The BSCFWY module allows the BSCTRAN, BSCFTI, and BSCRTI modules to work with Freeway without changing from previous versions of the product.

The ICPDLI module takes commands from the upper modules and makes the proper Freeway DLI calls. The DLI calls read and write information to Freeway and the ICP board. This module is similar to the one used for the DLI calls in the BSCDEMO program, described in the *BSCDEMO User's Guide*.

Chapter

2 BSCTRAN Commands and Options

Note

In this document, the term "Freeway" can mean either a Freeway server or an embedded ICP. For the embedded ICP, also refer to the user's guide for your ICP and operating system (for example, the *ICP2432 User's Guide for OpenVMS Alpha (DLITE Interface)*).

The BSCTRAN program accepts command strings interactively from a terminal or in batch mode by means of a DCL command procedure. BSCTRAN is invoked at the user's terminal by the following DCL command:

```
RUN BSCTRAN
```

The following prompt is then presented to the user:

```
BSC>
```

The following commands are available:

HELP	Display BSC commands and options.
SHOW	Display the current configuration of a BSC link.
SET	Establish the configuration of a BSC link.
DEFAULT	Enable or disable global options.
DIAL	Perform modem autodial and/or configuration.
SEND	Send a file to a remote BSC station.

RECEIVE Receive a file from a remote BSC station.

EXIT Terminate processing.

Comments are allowed on the BSCTRAN command line. Comments begin with an exclamation mark (!) and may be useful for clarifying the actions in a command procedure. See Chapter 7 for an example of a command procedure.

2.1 Specifying the Freeway Data Paths

BSCTRAN is a Simpact data link interface application (DLI for a Freeway server or DLITE for an embedded ICP). The BSCTRAN software calls DLI library functions. For a description of the DLI interfaces, refer to the *Freeway Data Link Interface Reference Guide* for the Freeway server. For the embedded DLITE interface, refer to the user's guide for your ICP and operating system (for example, the *ICP2432 User's Guide for OpenVMS Alpha (DLITE Interface)*).

The DLI configuration file defines "sessions." These are the data paths between the client application (in this case, BSCTRAN) and each ICP link. Using the session information, the Freeway software is able to route data from BSCTRAN to a specific ICP link on a specific ICP.

Simpact supplies special DLI and DLITE configuration files for use with BSCTRAN. These files are called `dlitrancfg` and `dlitetrancfg`, respectively. When you use BSCTRAN, you *must* use these as your configuration files. The files define data paths to many different links. By specifying the proper information in a BSCTRAN command, you can route data to any of the defined links. The following sections describe what information you need to specify in a BSCTRAN command to route data to a particular link.

2.1.1 Server Field in BSCTRAN Commands

Many of the BSCTRAN commands include a `server` parameter. Note that this parameter contains a *pointer* that allows the Freeway software to determine a DLI session name and thus a board number and a link number, by referencing the configuration files. Consider the following example. Suppose you want to use the `SHOW` command to display the configuration of a particular link on a particular ICP. To do this, use the `SHOW` command with the following parameters:

```
SHOW 2 FWYO_0
```

The "2" indicates link 2. The "0" following the underscore indicates ICP 0. The `FWYO` before the underscore is the `server` parameter (the first four characters of the DLI session name) that acts a pointer. Following is a description of how the BSCTRAN software uses this information to select the desired link and ICP board.

2.1.2 How the BSCTRAN Command Determines the Data Path

The BSCTRAN software combines the information in the BSCTRAN command into a session name as follows: the `server` parameter, followed by a "B," followed by the board number, followed by an "L," followed by the link number. Continuing the above example, the BSCTRAN software would combine the information in the `SHOW` command into the following session name: `FWYOB0L2`.

The software then checks the `dlitrancfg` (or `dlitetrancfg`) configuration file and locates the section that defines the designated session. In our example, it would locate the following session section:

```
FWYOB0L2
{
      .
    Transport = "bsc0";             // Connection Name  //
    BoardNo = "0";                  // ICP board number //
    PortNo = "2";                   // Link number      //
      .
```

For a detailed view of a sample Freeway server `dlitrancfg` file, refer to Section 7.1 on page 104. Note how in our example, the session definition properly indicates that the desired ICP board number is 0 and the desired link number is 2. Also note that the `Transport` parameter indicates a connection name of `bsc0`. The Freeway software uses the connection name to find the server name. To do so, the Freeway software checks the `tsitrancfg` file and locates the section that defines the designated connection. In our example, it would locate the following section:

```
bsc0
{
      .
    server = "freeway0";          // Server name //
      .
```

For a detailed sample `tsitrancfg` file (for a Freeway server), refer to Section 7.2 on page 107. Note how in our example, the connection definition properly indicates that the desired server name is `freeway0`.

For the embedded ICP boards (using DLITE), the Simpact-supplied DLI configuration file (`dlitetrancfg`) does not reference Simpact Freeway servers, and thus does not reference a TSI configuration file. The proper ICP and link are, however, defined exactly as in any DLI configuration file.

In summary, the BSCTRAN software combines the information in a BSCTRAN command into a session name and then looks up the definition of that session in the `dlitrancfg` or `dlitetrancfg` file. It uses the session definition to determine the desired link and ICP board. Using this information, the Freeway software can then route the data to the desired link on the desired ICP.

Figure 2–1 shows two example BSCTRAN commands and the way in which the information in the commands determines the data path. Note that for DLITE, the "`transport`" parameter is ignored in `dlitetrancfg`.

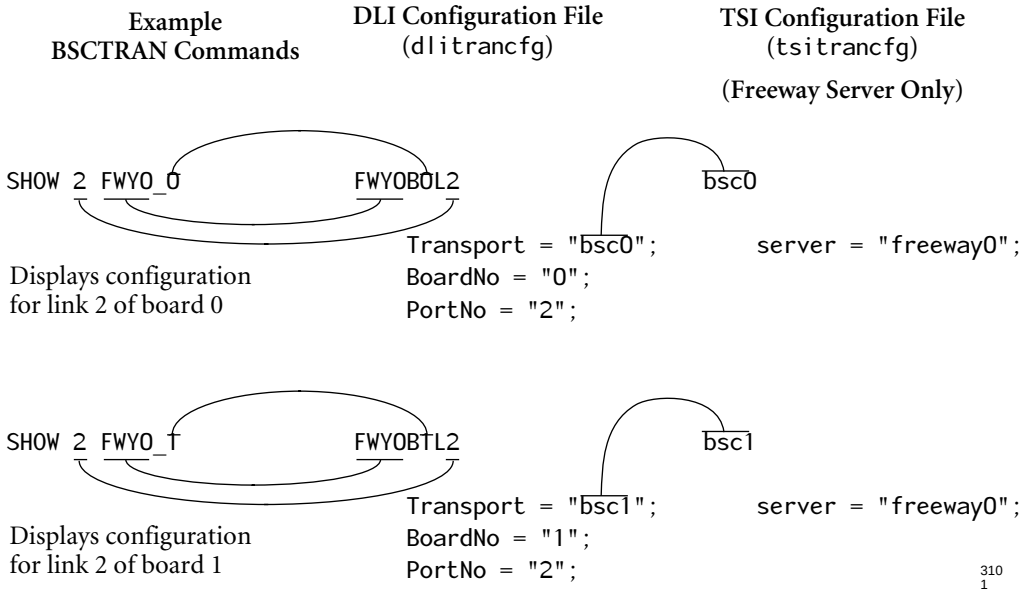

Figure 2–1: Example BSCTRAN Commands and Resulting Data Paths

Note

BSCTRAN is designed to have sessions active on only one ICP at a time. When you use any command to access another ICP or another Freeway server, all connections to the previous ICP are automatically closed and any active links are "stopped."

2.1.3 Important Considerations for Freeway Server

Note the following important considerations before using BSCTRAN in a Freeway server. These notes assume that you have read the previous sections.

- Check to make sure that the supplied DLI and TSI configuration files, `dlitrancfg` and `tsitrancfg`, are set up to meet the needs of your system. Note especially:
  - If your Freeways are not named `freeway0` and `freeway1`, you need to supply the correct server names in the `tsitrancfg` file (see Section 7.2 on page 107).
  - If you have more than two Freeways, you need to add appropriate session definitions to the `dlitrancfg` file, and appropriate connection definitions to the `tsitrancfg` file.
- When using the BSCTRAN commands, you must use the appropriate server pointers in the `server` parameter. The default server pointers are `FWY0` and `FWY1` which point to the default server names of `freeway0` and `freeway1`, respectively.
- You can use values other than `FWY0` and `FWY1` for the server pointers in BSCTRAN commands. However, if you do so, you need to modify the session names in the `dlitrancfg` file. For example, if you want to use `SERVERX` instead of `FWY0` in the BSCTRAN commands, you need to change session names as follows: `FWY0BOL0` to `SERVERXBOL0`, `FWY0BOL1` to `SERVERXBOL1`, and so on. The server pointers must be 4 to 10 characters in length.
- If you modify the `dlitrancfg` or `tsitrancfg` files, you must regenerate the binary versions of the files (`dlitrancfg.bin` and `tsitrancfg.bin`). For a detailed description DLI and TSI configuration files, see the *Freeway Data Link Interface Reference Guide* and the *Freeway Transport Subsystem Interface Reference Guide*
- Do not change the names of the `dlitrancfg` and `tsitrancfg` files.

2.1.4 BSCTRAN Logging

2.1.4.1 VMS Logical Names and Logging

BSCTRAN supports the use of VMS logical names for the link number and server name. The maximum length of a logical name is 10 characters. A logical name may be defined in any VMS logical name table that is included in the LNM$FILE_DEV logical name table search list. (This list usually includes the process, job, group, and system tables.) If a server logical name is used, it will be prefixed to all log messages. Log messages are output to SYS$OUTPUT and optionally to an operator console and disk log file. Log messages have the following format:

```
LLLLLLLLL_SSSSSSSSS_N: DD-MMM-YYYY HH:MM:SS.CC  Message Text
```

where:

LLLLLLLLL	=	optional VMS logical server name and ICP number
SSSSSSSSS	=	server parameter
N	=	link number

An example of using a logical name defined in the process table is:

```
$ DEFINE SIMPACT FWYO_0
$ DEFINE LINK1 2
$ RUN BSCTRAN
BSC> SEND TXT.TEST LINK1 SIMPACT
SIMPACT_FWYO_0_2: 10-APR-1994 10:40:08.78 Sending       TXT.TEST
SIMPACT_FWYO_0_2: 10-APR-1994 10:40:09.40 Send Complete TXT.TEST=18
Records
```

2.1.4.2 UNIX Logging

There is no UNIX logical name support, and BSCTRAN log messages have a UNIX-style time and data format:

```
                Day Mon DD HH:MM:SS YYYY
For example:    Sun Dec 31 23:59:59 1995
```

2.2 HELP Command

The HELP command displays the format for all legal BSCTRAN commands. It takes no argument. Figure 2–2 is an example of the HELP command. In the figure, square brackets [ ] are used to indicate optional fields for commands. After the device name and link number have been established, the use of brackets is optional on subsequent commands.

Note

The DEFAULT command options [/[NO]PRINT], [/[NO]BATCH], and [/[NO]LOG[=n]] do not apply to UNIX.

```
BSC> HELP
     SIMPACT BSC File Transfer Program BSCTRAN [V03-06]
     SHOW link# server[_ICP#] [/DEFAULT] [/STATS] [/CLEAR]
     SET link# server[_ICP#] [/START or /LOCKSTART] [/STOP] [/PARn[=d]]
     SEND filename link# server[_ICP#] [/DTIME=h:m:s] [/BINARY]
               [/TRANS] [/CARD] [/NOTEOF]
     RECEIVE filename link# server[_ICP#] [/FIXED] [/BINARY] [/RECL=n]
               [/FORTRAN] [/ESCAPE] [/DTIME=h:m:s]
     DEFAULT   [/FILE=filename]
               [/FIXED] [/BINARY] [/RECL=n] [/ESCAPE] [/FORTRAN]
               [/[NO]PRINT] [/[NO]BATCH] [/[NO]LOG[=n]]
               [/[NO]DLOG] [/[NO]DELETE] [/[NO]SAFE] [/[NO]ABORT]
               [/RECORDS=[2780,3780]] [/BUFSIZE=n]
     DIAL link# server[_ICP#] [/STRING=s] [/CONFIG=s]
               [/RESPONSE] [/DTIME=h:m:s]
     EXIT
```

Figure 2–2: Example of the HELP Command

Note

The `server` parameter is not the server name, but is a pointer to the server name. ***It is important that you read Section 2.1 on page 20 to understand the `server` parameter.***

Note that the ICP number is appended to the `server` parameter using an underscore character. If there are multiple boards, the ICP number determines which ICP board to connect to. If desired, a dash (–), forward slash (/), or equal sign (=) can be used in place of the underscore. If you omit the ICP number after the `server` parameter, BSCTRAN will use a default ICP number of 0. The link number is still used to select an individual serial link on the ICP board.

The `server` parameter and ICP number appear in log messages that are output to the system output device and optionally to a disk log file and VMS operator console. See Section 2.5 on page 33 regarding optional logging. Log messages have the following format:

```
VMS Format:     SSSSSSSSSS_B_N: DD-MMM-YYYY HH:MM:SS.CC  Message Text
UNIX Format:    SSSSSSSSSS_B_N: Day Mon DD HH:MM:SS YYYY Message Text
```

where:

`SSSSSSSSSS`	=	`server` parameter
`B`	=	ICP board number
`N`	=	link number

2.3 SHOW Command

The SHOW command has the following format:

```
SHOW link# server[_ICP#] [/DEFAULT] [/STATS] [/CLEAR]
```

The SHOW command displays the current configuration of the link (link#) on the server pointed to by the server parameter. ***It is important that you read Section 2.1 on page 20 to understand the* `server` *parameter.*** Table 2–1 is an example of the SHOW command output indicating the default BSC link configurations immediately after downloading the BSC software to the ICP.

Table 2–1: SHOW Command Output

Link Configuration
Local Link 0 Status = INACTIVE

ICP Buffer Size = 1024 **104 buffers free out of 104**

Par #	Description	Value	Par #	Description	Value
1	Data rate	9 (9600)	2	Clock source	0 (external)
3	Reply timer length	3	4	Number syncs	3
5	Protocol	0 (3780)	6	Parity	1 (odd)
7	Character set	1 (EBCDIC)	8	Transmit block size	512
9	Record separator	0 (none)	10	Data translation	1 (Table 1)
11	Station priority	0 (slave)	12	Space compression	0 (disabled)
13	Conversation mode	0 (disabled)	14	Retry limit	3
16	Modem control	1 (full duplex)	17	Safe store	0 (disabled)
19	Message blocking	1 (data)	20	Block checking	1 (exclude first)
22	EOM line control	0 (reverse)	22	Data Ack node	0 (disabled)
24	Alternating Ack	1 (enabled)	26	TTD/WACK	1 (enabled)
28	RVI handling	0 (continue)	30	DSR delay	3
35	Modem type	1 (SADL)			

See the *BSC Programmer's Guide* for a complete description of ICP link configuration parameters.

The SHOW command options are as follows:

/DEFAULT This option displays the current values of the DEFAULT command options. Because these are global options, no link# or server parameter is required for this command. The following is the output from the SHOW /DEFAULT command upon startup of the BSCTRAN program:

```
BSC> SHOW /DEFAULT

          BSCTRAN Write Buffer Size = 1024
          Default Receive Filename  = DEFAULT.DAT /RECL=254
          PRINT Received Files      = DISABLED
          BATCH Received Files      = DISABLED
          LOGGING of Errors/Stats   = DISABLED
          DISK LOGGING Errors/Stats = DISABLED
          DELETE Partial Recv Files = DISABLED
          SAFE STORE on Receive     = DISABLED
          ABORT after Fatal Errors  = DISABLED
          RECORD Handling           = 3780 (RS)
```

Note

The "PRINT Received Files," "BATCH Received Files," and "LOGGING of Errors/Stats" do not apply to the UNIX display.

/STATS This option displays the current values of an ICP Link Statistics Report.

/CLEAR This option displays the current values of an ICP Link Statistics Report, then sets all link statistics to zero. The /CLEAR option must be used with the /STATS option. The following are outputs for the /STATS and /CLEAR options of the SHOW command:

```
BSC> SHOW 2 spot_1 /STATS /CLEAR ! Statistics before clearing

      CURRENT STATISTICS FOR LINK #2:
Recv data messages  = 2 Xmit data messages = 4
Recv blocks         = 9 Xmit blocks        = 12
Recv NAKs           = 0 Xmit NAKs          = 1
Recv buffer errors  = 0 Xmit buffer errors = 0
Recv BCC errors     = 0 Recv parity errors = 0
Recv overrun errors = 0
BSC> SHOW /STATS       ! The statistics have been cleared

      CURRENT STATISTICS FOR LINK #2:
Recv data messages  = 0 Xmit data messages = 0
Recv blocks         = 0 Xmit blocks        = 0
Recv NAKs           = 0 Xmit NAKs          = 0
Recv buffer errors  = 0 Xmit buffer errors = 0
Recv BCC errors     = 0 Recv parity errors = 0
Recv overrun errors = 0
```

2.4 SET Command

The SET command has the following format:

```
SET link# server[_ICP#] [/PARn[=d]] [/PARn[=d]...]
                [/START] [/STOP]
```

The SET command sets up the configurations of the link (link#) on the ICP board (ICP#).

The SET command options are as follows:

/PARn[=**d**] This option specifies the value of configuration parameter n, where n = 1 through 30, and d = 0 through the maximum allowed value for the particular parameter n. If d is omitted, BSCTRAN will prompt the user with the allowed values. For a complete description of the link configuration parameters, see the *BSC Programmer's Guide*.

/START This option starts the specified link. The START option may be used with the /PARn=d options. A link must be started before attempting to send or receive files. An attempt to start a link that is already enabled will be rejected. If the error Link not started is reported by BSCTRAN, and a SHOW command indicates the link status as STARTING, this means that the Data Set Ready (DSR) signal has not been received from the remote computer. This error can be suppressed by using the SET /PAR16=2 or SET /PAR16=3 command to instruct the ICP to ignore the DSR signal.

/LOCKSTART This option is identical to the /START option with the exception that the link stays enabled until a SET /STOP (disable) command is sent, even if the disable is not sent by the current process.

Caution

Be aware of the implications of using the SET /LOCKSTART command. After a link is enabled with /LOCKSTART, it remains enabled (even if BSCTRAN terminates) until the link is disabled with a SET /STOP command. A link that remains enabled after termination of a BSCTRAN process (due to a LOCKSTART) continues receiving messages into its data queues until it either runs out of buffers, or until a new BSCTRAN process references (attaches) the same link. As soon as an attach is made, data is sent to BSCTRAN. Therefore, in order to receive data accumulated on a link, the first command must be RECEIVE. Any other command discards received data that is not relevant to the command. For example, a SHOW command discards any data packets that are not responses to its report requests.

/STOP This option stops the link (link#). A link must be stopped before issuing a SET /PARn=d command. Note that when the BSCTRAN program is exited, all active links are automatically stopped.

The following commands show how to set the data rate (/PAR1) directly, obtain a help prompt for character set translation (/PAR7), and start the link:

```
BSC> SET 2 spot_1 /STOP
BSC> SET /PAR1=9/PAR7/START
PAR7(Char Set Trans) 0=ASCII 1=EBCDIC 2=ASC/CRC (0 to 2): 1
```

Note

The message "ERROR: Link not started" after an initial SET /START command is normal until the second link is up.

Note

The message "WARNING: Link already started" is possible if a previous BSCTRAN process started the link with SET /LOCKSTART.

2.5 DEFAULT Command

The DEFAULT command has the following format:

```
DEFAULT   [/FILE=filename]
          [/FIXED] [/BINARY] [/RECL=n] [/ESCAPE] [/FORTRAN]

          [/[NO]PRINT] [/[NO]BATCH] [/[NO]LOG[=n]]
          [/[NO]DLOG] [/[NO]DELETE] [/[NO]SAFE] [/[NO]ABORT]
          [/RECORDS=[2780,3780]] [/BUFSIZE=n]
```

Note

The DEFAULT command options [/[NO]PRINT] [/[NO]BATCH] [/[NO]LOG[=n]] do not apply to UNIX.

The DEFAULT command allows selective enabling and disabling of the global options which remain in effect until changed or until the BSCTRAN program exits.

The DEFAULT command options are as follows:

/FILE=filename This option specifies the filename to receive any unexpected data. There are five suboptions for the /FILE option:

- /FIXED
- /BINARY
- /RECL=n
- /ESCAPE
- /FORTRAN

These suboptions have the same meanings as they do for the RECEIVE command explained in Section 2.8. The default file name and attributes upon BSCTRAN startup are DEFAULT.DAT /RECL=254.

/[NO]PRINT If this VMS option is enabled, each received file will be automatically queued to the system printer after the file is closed. The default is /NOPRINT.

/[NO]BATCH If this VMS option is enabled, each received file will be automatically queued to the system batch queue after the file is closed. The default is /NOBATCH.

/[NO]LOG[=n] If this VMS option is enabled, all errors, warnings, and transmission statistics will be logged to the operator console designated by n, where n is a value from 1 through 12 representing VMS OPER1 through OPER12 (default is 1 for OPER1). The default setting for this option is /NOLOG.

For example, to enable logging to an operator console, the following DCL command is first performed at the operator console:

```
$ REPLY/ENABLE=OPER2
```

Then the following BSCTRAN commands will cause all log messages to be sent to the OPER2 operator console:

```
$ RUN BSCTRAN
BSC> DEFAULT /LOG=2
BSC> SHOW /DEFAULT
  BSCTRAN Write Buffer Size = 1024
  Default Receive Filename  = DEFAULT.DAT /RECL=254
  PRINT Received Files      = DISABLED
  BATCH Received Files      = DISABLED
  LOGGING of Errors/Stats   = ENABLED to OPER2
  DISK LOGGING Errors/Stats = DISABLED
  DELETE Partial Recv Files = DISABLED
  SAFE STORE on Receive     = DISABLED
  ABORT after Fatal Errors  = DISABLED
  RECORD Handling           = 3780 (RS)
BSC>
```

/[NO]DLOG If this option is enabled, all errors, warnings, and transmission statistics will be logged to a disk log file. The default is `/NODLOG`. The disk log file name has the following format:

```
SSSSSSSSSS_B_N_MMMDD.LOG
```

where:

`SSSSSSSSSS`	=	`server` parameter. ***It is important that you read Section 2.1 on page 20 to understand this parameter.***
`B`	=	ICP board #
`N`	=	Link #
`MMMDD`	=	Month/Day

Each different device/link pair will have a separate log file. Log messages are appended to the log file until midnight. A new log file is created when the date changes.

/[NO]DELETE If this option is enabled, any received file that is incomplete due to error will be deleted and a warning message logged. The default setting, `/NODELETE`, saves incomplete received files and logs a warning message indicating that a partial file was received.

/[NO]ABORT If this option is enabled, BSCTRAN will stop the link and abort with `LIB$STOP` upon receiving any of the following errors:

- DSR Timeout
- Disconnect
- EOT received
- Retry limit
- RVI received during `SEND`
- Autodial failure
- BSCTRAN timeout (read or write)

The abort is performed at the BSCFTI level so that users writing their own applications can control this option by calling the BSCDFAULT procedure. The default is `/NOABORT`.

/[NO]SAFE If this option is enabled, BSCTRAN will send a Safe Store Acknowledgment to the ICP upon successful closing of any received file. If the file close is unsuccessful, BSCTRAN will send a Safe Store Negative Acknowledgment. If any other error occurs during receive, such as timeout, no safe store action will be performed. The default is `/NOSAFE`.

Caution

The user is responsible for setting Safe Store *on* for the link. If the link is not configured correctly, the Safe Store Acknowledgment will be rejected.

/RECORD=[2780,3780] BSCTRAN uses this option to determine whether to insert a Record Separator character (if `3780` is set) or a Unit Separator character (if `2780` is set) after each non-transparent record sent to the ICP. This option has no effect on BSCTRAN's handling of records received from the ICP. The default is `/RECORD=3780`.

The following steps send and receive a file in 2780 mode, first as transparent 2780, then as non-transparent 2780:

```
$ RUN BSCTRAN
BSC> SET 2 spot_1 /START          ! Start link 2 on server "spot"
BSC> SET 3 /START                 ! Start link 3 on server "spot"
BSC> DEFAULT /RECORD=2780         ! Set BSCTRAN for 2780 records
BSC>SEND X2780.TEST 2 /TRANS      ! Send Transparent 2780
BSC> RECEIVE X2780.RECV 3 /BIN    ! Use /BIN to receive transparent
BSC>SEND 2780.TEST 2              ! Send non-transparent 2780
BSC> RECEIVE 2780.RECV 3          ! Don't use /BIN for non-transparent
```

To send transparent 2780 records, BSCTRAN prefixes each record with the count. Records are accumulated into the BSCTRAN write buffer without splitting records. Blocks are written to the ICP as command codes 22 and 23. The ICP transmission block should be equal to or greater than the maximum record size + 2.

Note

The `SEND /CARD` option is not supported for transparent 2780.

To send non-transparent 2780 records, BSCTRAN handles records in the same way as 3780, except that a Unit Separator character (hex 1F) is inserted at the end of each record, rather than a Record Separator (hex 1E).

In receive mode, BSCTRAN record handling is determined by the command code. Command codes 22 and 23 signify transparent 2780 records, each prefixed with a count. Command codes 20 and 21 signify transparent data, usually 3780, but can also be single-record transparent 2780 blocks. Command codes 16 and 17 signify non-transparent data, either 2780 or 3780. BSCTRAN determines record boundaries by the RS or US characters.

Refer to the *BSC Programmer's Guide* for more information on 2780 and 3780 message blocking.

/BUFSIZE=n This option controls the size of the BSCTRAN write buffer and the ICP message buffers. The default value for this option is 1024; the maximum value is 4096.

Cautions:

1. The default ICP message buffer size for BSCTRAN is 1024 bytes. When the BSCTRAN program is run, it sends a buffer configuration command to set the same size buffers on the ICP, only if this size is changed from 1024 with the `/DEFAULT` command. There are two other buffer pools that must be configured for BSCTRAN to work correctly. One buffer pool is used with the Freeway DLI routines. The size of these buffers is set in the DLI configuration file. The second buffer pool is the Freeway server buffers (if the Freeway server DLI is used). The size of these buffers is set in the TSI configuration file. Both these buffer sizes need to be set to a size that includes the BSCTRAN buffer size and TSI header size. For example, if the BSCTRAN buffer size is 1024 bytes, the DLI and TSI buffers should be configured to 1200 bytes.

2. Users who wish to force larger blocks of data to be transmitted on the BSC communications link should use the following commands after all links have been disabled:

```
$ RUN BSCTRAN
BSC> DEFAULT /BUFSIZE=2048      ! Maximum is 4096
BSC> SET 2 spot /PAR8=2050      ! Transmission block size
BSC> SHOW 2                     ! check your settings
```

   The transmission block size should be `BUFSIZE` + 2 to allow for the STX and ETX characters which the ICP inserts.

3. It is not necessary to change the transmission block size unless the user wants to force blocks of an exact size on the BSC communications link. The ICP transmission block size is independent of both the ICP message buffer size and the BSCTRAN write buffer size.

4. A `BUFSIZE` of less than 1024 is not recommended.

2.6 DIAL Command

The DIAL command has the following format:

```
DIAL link# server[_ICP#] [/STRING=s] [/CONFIG=s]
                [/RESPONSE] [/DTIME=h:m:s]
```

The DIAL command sends the autodial and/or modem configuration strings to the ICP to communicate with the modem type specified by ICP configuration option 35 for performing autodial operations on the link (link#) of the ICP board (ICP#). The link must be disabled or the DIAL command will be rejected.

BSCTRAN will display the state changes of the autodial operations with the following statements, as well as applicable success, failure, or modem response messages. The messages will also be logged as specified by the DEFAULT /LOG or /DLOG options.

```
spot_1_2: 15-NOV-1990 14:21:13.11  MODEM OPTION:  <configuration string>
spot_1_2: 15-NOV-1990 14:21:15.92  MODEM DIAL:  <dial string>
```

The DIAL command options are as follows:

/STRING=s This option sends an Autodial Start command (command code 38) to the ICP, where s is the phone number string, including any command characters appropriate for the modem specified by ICP configuration option 35. Refer to the *BSC Programmer's Guide* for details on the support of various modem types.

/CONFIG=s This option sends a Modem Configuration command (command code 39) to the ICP, where s is the configuration string appropriate for the modem specified by ICP configuration option 35. Refer to the *BSC Programmer's Guide* for details on the support of various modem types.

/RESPONSE If this option is used, all responses from the modem will be logged to the system output device, as well as any log files specified by the DEFAULT /LOG or /DLOG options.

/DTIME=h:m:s This option is the timeout specification (default time is 1 minute). This is the maximum time that BSCTRAN will wait for the ICP to return the modem response. If timeout occurs, BSCTRAN will abort or return the user to the BSC prompt, depending on the setting of the global `DEFAULT /ABORT` option.

2.7 SEND Command

The SEND command has the following format:

```
SEND filename link# server[_ICP#] [/DTIME=h:m:s]
                          [/BINARY]
                          [/TRANS] [/CARD] [/NOTEOF]
```

The SEND command sends the file (filename) for transmission on the link (link#) of the ICP board (ICP#).

Upon receiving a SEND command, the BSC File Transfer program will open the file, accumulate the file records into blocks (maximum block size determined by the DEFAULT /BUFSIZE option) without splitting records, and send the blocks to the ICP for transmission to the remote station.

BSCTRAN will display the state changes of the file transfer with the following statements:

```
spot_1_2: 10-APR-1989 10:40:08.78  Sending        TXT.TEST
spot_1_2: 10-APR-1989 10:40:09.40  Send Complete  TXT.TEST=18 Records
```

The SEND command options are as follows:

/DTIME=h:m:s This option is the timeout specification (default time is 1 minute). This is the maximum time that BSCTRAN will wait for the ICP to respond with a data acknowledgment after each block is sent. If timeout occurs, BSCTRAN will abort or return the user to the BSC prompt, depending on the setting of the global DEFAULT /ABORT option.

/BINARY This option sends each record separately to the ICP, without inserting a record separator character. This option is normally used with the /TRANS option for sending transparent 3780 data.

Caution

If the user needs to ensure that binary data records (sent without the `/TRANS` option) are not concatenated when sent by the ICP, message blocking should be set to *off*.

/TRANS

This option specifies that BSCTRAN will instruct the ICP to send the file in transparent mode. This option is necessary to send files such as object or executable images which may contain control characters that might be misinterpreted by the BSC communication protocol. The BSC 2780/3780 protocol performs translation of transparent data according to the setting of the data translation configuration parameter. This option is normally used with the `/BINARY` option to send transparent 3780 data. For transparent 2780 data, only the `/TRANS` option should be used.

Cautions:

1. The ICP must be configured to the EBCDIC character set (or ASCII with no parity) for the `/TRANS` option to work properly. Also, *Data Translation* (option 10) must be set to *off* when using EBCDIC to prevent the ICP from translating the transparent text.

2. The ICP will send transparent data as if *Message Blocking* (configuration option 19) is set to *off*. That is, each message buffer from the client is sent as a single transmission block on the link, assuming the transmission block is large enough.

   The transmission block, therefore, should be at least the size of the largest record + 2 to prevent splitting of larger records. This proper combination of `/TRANS` and the appropriate transmission block size ensures that transparent record boundaries will be preserved.

3. Refer to the *BSC Programmer's Guide* for information regarding the relationship between transparent data and the ICP data translation setting.

/CARD This option sends the file in 80-character card image format. Records are split or padded with spaces if necessary. Many IBM systems require incoming data to be in this format. This option is not supported for transparent 2780 records.

/NOTEOF This option specifies that file transmission ends with an ETB rather than an ETX block. This is useful for sending multiple files as one concatenated file to the receiving system. The last file to be concatenated is sent without the `/NOTEOF` option.

2.8 RECEIVE Command

The `RECEIVE` command has the following format:

```
RECEIVE filename link# server[_ICP#] [/DTIME=h:m:s] [/BINARY]
                             [/RECL=n] [/FIXED]
                             [/FORTRAN] [/ESCAPE]
```

The `RECEIVE` command creates the file (`filename`), receives records from the link (`link#`) on the ICP (`ICP#`), and writes them into the file until the last record (the ETX block) is received. Then the file is closed.

The BSCTRAN program recognizes Punch and Print component select sequences at the beginning of the first record in all data blocks received from the ICP. If detected, these sequences are deleted from the incoming data.

Valid Punch select sequences are:

- ESC 4
- DC2
- DC3

Valid Print select sequences are:

- DC1
- ESC X (where X is anything except 4)

A Print select sequence of ESC X is treated as a forms-control sequence; ESC is deleted and X is the forms-control character. If neither the `/FORTRAN` nor `/ESCAPE` option has been specified, a default will be determined based on the value of X. Refer to the `/FORTRAN` option on page 46 and the `/ESCAPE` option on page 47.

The BSC File Transfer program displays the state changes of the file transfer with the following statements:

```
spot_1_3: 10-APR-1989 10:40:09.68  Receiving      TXT.RCV
spot_1_3: 10-APR-1989 10:40:09.88  Recv Complete  TXT.RCV=18 Records
```

The RECEIVE command options are as follows:

/DTIME=h:m:s This option is the timeout specification (default time is 1 minute). This is the maximum time BSCTRAN will wait for the data records from the ICP. If timeout occurs, BSCTRAN will abort or return the user to the BSC prompt, depending on the setting of the global DEFAULT /ABORT option.

/BINARY This option specifies that BSCTRAN not search for record separators. Records are deblocked by the /RECL size (default = 512 bytes). The /BINARY option should be used to receive files that were sent in either 2780 or 3780 transparent mode. This option is not compatible with either the /FORTRAN or /ESCAPE forms-control option.

Caution

If the received binary records were not sent as transparent data, message blocking should be set to *off* to prevent the records from being concatenated when received by the ICP. This will preserve the record boundaries.

/RECL=n This option specifies the file record length, where n = number of bytes. The default is 512 and the maximum is 4096. For variable record format (which is the default format), this argument becomes the maximum length. RECL should be equal to (or greater than) the maximum record size of the file being received.

/FIXED This option specifies that the file is opened with fixed-length record format, where the length is determined by the `/RECL` option (default = 512). If incoming records are not already in the specified fixed-length format, BSCTRAN will continue reading records from the ICP, concatenating and splitting records as necessary to accumulate the required length. The last record will be padded with ASCII spaces if necessary. This option (combined with the `/BINARY` option) should be used when receiving an executable-image file from another site. `/FIXED` is not compatible with either the `/FORTRAN` or `/ESCAPE` forms-control option.

/FORTRAN If this option is specified, the BSC File Transfer program will look for Fortran forms-control characters at the beginning of each incoming data record. If the first character is an escape character, the second character is interpreted as Fortran forms control, and the escape character is deleted. Otherwise, the first character of the record is interpreted as the Fortran forms-control character.

The Fortran forms-control character is converted to the equivalent RMS (record management system) print file carriage control. Valid characters are:

Character	Meaning
space	Single space
0	Double space
1	Form feed
+	Overprint
$	Prompt
All others	Single space

/ESCAPE

If this option is specified, the BSC File Transfer program looks for a two-character forms-control escape sequence at the beginning of each incoming data record. The escape character is deleted and the forms control character is converted to the equivalent RMS print file carriage control. The escape sequence must consist of an escape character followed by one of the following:

Character	Meaning
/	Single space
S	Double space
T	Triple space
M	Overprint
A	Form feed
All others	Single space

If an escape sequence is found in the first data record (and neither the `/FORTRAN` nor the `/ESCAPE` option is specified), forms control will default to `/FORTRAN` or `/ESCAPE` depending on the character following the escape character.

In `RECEIVE` mode, the BSC File Transfer program will open the file with defaults of record format = variable, record attribute = carriage return carriage control, and maximum record length = 512, unless directed otherwise by certain options. Table 2–2 shows the effect of various options on record format and attributes.

Table 2–2: RECEIVE File Open Options

BSCTRAN Option	RMS Record Format	RMS Record Attributes
(Default)	Variable	Carriage return carriage control
`/FIXED`[a]	Fixed	None
`/BINARY`	Variable	None
`/FORTRAN`	VFC[b]	Print file carriage control
`/ESCAPE`	VFC	Print file carriage control

[a] `/FIXED` takes precedence over `/BINARY` for record format.

[b] VFC record format = variable with fixed-length control.

2.9 EXIT Command

The EXIT command has the following format:

```
EXIT
```

The EXIT command causes the BSC File Transfer program to exit and return the user to the system command level. All active links are disabled with the EXIT command.

Chapter

3 File Transfer Interface (BSCFTI)

Note

In this document, the term "Freeway" can mean either a Freeway server or an embedded ICP. For the embedded ICP, also refer to the user's guide for your ICP and operating system (for example, the *ICP2432 User's Guide for OpenVMS Alpha (DLITE Interface)*).

The BSC File Transfer Interface consists of a set of procedure calls that are used by the BSC User Interface (BSCTRAN) to send and receive files from Freeway. You may use these calls from your own application by linking your program with the modules `BSCFTI.OBJ` and `BSCRTI.OBJ`.

The BSC File Transfer Interface is the easiest interface for application programmers to use. It performs all the error handling related to unexpected responses from Freeway. Error handling is explained in detail in Chapter 5.

This chapter describes each BSCFTI procedure, and its associated parameters and return codes. See Table 5–2 on page 95 for a complete list of the BSC File Transfer program error codes.

Section 3.9 on page 70 shows two example programs calling BSCFTI routines from languages other than C.

3.1 BSCDFAULT Procedure

Calling Sequence

```
BSCDFAULT (fname, fixed, binary, fortran, escape,
          recl, bufsize, print, batch, log, dlog,
          delete, abort, safe, rec3780)
```

Note

The `print`, `batch`, and `log` arguments of the `BSCDEFAULT` procedure do not apply to UNIX.

Procedures Called

None

Functions

1. Reconfigures the BSCRTI global `bscdefault` structure (refer to the C typedef BSCDFLT structure in the `BSCTRAN.INC` include file). The default values for this structure are described in Section 2.5 on page 33.

Arguments

fname This is a pointer to a NULL-terminated character string. It is the default filename to be used when unexpected data is received. Parameters `fixed`, `binary`, `fortran`, `escape`, and `recl` determine the attributes of the default file.

fixed This is a pointer to an integer. It is a longword containing the record format indicator. If the indicator is true, the default file is opened with fixed-length record format.

binary This is a pointer to an integer. It is a longword containing a boolean flag. If the flag is true, received data records are *not* deblocked using

record separator characters, but by `recl` length only. A non-zero value is treated as true.

fortran This is a pointer to an integer. It is a longword containing a boolean flag. If the flag is true, BSCRECV will look for Fortran forms-control characters at the beginning of each incoming record. A non-zero value is treated as true.

escape This is a pointer to an integer. It is a longword containing a boolean flag. If the flag is true, BSCRECV will look for a two-character forms-control escape sequence at the beginning of each incoming record. A non-zero value is treated as true.

recl This is a pointer to an integer. It is a longword containing the maximum record length of the default file. A length of up to 4096 bytes is supported.

bufsize This is a pointer to an integer. It is a longword containing the BSCTRAN write buffer size, which will also be the ICP message buffer size if BSCDFAULT is called when all links are disabled and no ICP message buffers are in use.

print Under VMS, this is a pointer to an integer. It is a longword containing a boolean flag. If the flag is true, received files will automatically be queued to the system printer. A non-zero value is treated as true.

batch Under VMS, this is a pointer to an integer. It is a longword containing a boolean flag. If the flag is true, received files will automatically be queued to the system batch queue. A non-zero value is treated as true.

log Under VMS, this is a pointer to an integer. It is a longword containing the operator console number (valid values are 1 through 12 for VMS

OPER1 through OPER12). Log messages will be output to the operator console. A zero value will disable operator console logging.

dlog This is a pointer to an integer. It is a longword containing a boolean flag. If the flag is true, log messages will be output to a disk log file. A non-zero value is treated as true.

delete This is a pointer to an integer. It is a longword containing a boolean flag. If the flag is true, incomplete received files will be deleted; otherwise, they will be saved. A non-zero value is treated as true.

abort This is a pointer to an integer. It is a longword containing a boolean flag. If the flag is true, procedure BSCFTI will disable the link and abort with `LIB$STOP` after certain fatal errors (refer to Section 2.5 on page 33). A non-zero value is treated as true.

safe This is a pointer to an integer. It is a longword containing a boolean flag. If the flag is true, procedure BSCRECV will send a Safe Store Acknowledgment after a received file is successfully closed, or a Safe Store Negative Acknowledgment if the file close was unsuccessful. A non-zero value is treated as true.

rec3780 This is a pointer to an integer. It is a longword containing a boolean flag. If the flag is true, procedure BSCSEND will separate outgoing records with a 3780 record separator character (ASCII 30). Otherwise, a 2780 unit separator character (ASCII 31) is used. A non-zero value is treated as true.

3.2 BSCDIAL Procedure

Calling Sequence

```
ReturnCode = BSCDIAL (lnum, server, dial, config, dtime,
                      response, display)
```

Procedures Called

BSCDAS, BSCASL, BSCMODEM, BSCRESET, and BSCFWY

Functions

1. Verifies legal parameters.
2. If there is a change in the server name or ICP device, calls BSCASL to assign input and output channels.
3. If the `config` descriptor length is non-zero, sends the Modem Configuration command (command code 39) to the ICP link (`lnum`) with the configuration string.
4. If the `dial` descriptor length is non-zero, sends the Autodial Start command (command code 38) to the ICP link (`lnum`) with the dial string.
5. Issues a read to the ICP and waits up to `dtime` for a response.
6. Logs autodial success or error messages based on the `DEFAULT /LOG=n /DLOG` options and the `display` parameter.
7. If the `response` parameter is true, logs the modem response string.

Arguments

lnum This is a pointer to an integer. It is a longword containing the link number.

server This is a pointer to a descriptor. It is a character string containing the server name. An ICP board number may be appended to the server name with an underscore character; otherwise, the ICP board number is assumed to be 0. The maximum length is ten characters.

dial This is a pointer to a descriptor. It is a character string containing the autodial string. The maximum length is 256 characters.

config This is a pointer to a descriptor. It is a character string containing the modem configuration string The maximum length is 256 characters.

dtime This is a pointer to a descriptor. It is a character string containing the timeout value in the following time string format:

VMS format:	`dddd hh:mm:ss`
UNIX format:	`hh:mm:ss`

response This is an integer. It is a longword containing the boolean flag. If the flag is true, BSCDIAL logs the modem response string.

display This is an integer. It is a longword containing the boolean flag. If the flag is true, BSCDIAL logs messages to the system output device.

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success RCOK

Failure RCILPAR, RCFAIL, RCREJECT
Failure codes from BSCDAS, BSCASL, BSCMODEM
Unexpected ICP response failure codes (Table 5–3 on page 98)

3.3 BSCLOG Procedure

Calling Sequence

```
BSCLOG (lnum, display, message)
```

Procedures Called

None

Functions

1. Builds current device name, link number (`lnum`), system time, and text of `message` into a log message.
2. Checks BSCTRAN global options (maintained by procedure BSCDFAULT) for operator console and disk file logging. If enabled, these functions are performed.
3. If `display` is true, outputs log message to the system output device.

Arguments

lnum This is a longword containing the link number.

display This is a longword containing a boolean flag. If the flag is true, log messages will be output to the system output device. A non-zero value is treated as true.

message This is a pointer to a string. It is a NULL-terminated character string containing the text of the message to be logged. The maximum length is 210 characters.

3.4 BSCRECV Procedure

Calling Sequence

```
ReturnCode = BSCRECV (fname, lnum, server, dtime, fixed,
                      recl, binary, forms, display)
```

Procedures Called

BSCDAS, BSCASL, BSCREAD, and BSCRESET

Functions

1. If there is a change in the server name or ICP device, calls BSCASL to assign input and output channels.
2. Opens the file (`fname`) with the specified format.
3. Calls BSCREAD to get a record from the ICP link (`lnum`).
4. Performs forms-control translation if required.
5. Writes the record to the file.
6. Continues reading and writing records until error, timeout (`dtime`), or end-of-file is returned.
7. Closes the file.
8. Logs the transmission statistics.

Arguments

fname This is a pointer to a descriptor. It is a character string containing the file specification of the file to receive data blocks from the ICP. The maximum length is 255 characters.

lnum This is a pointer to an integer. It is a longword containing the link number.

server This is a pointer to a descriptor. It is a character string containing the server name. An ICP board number may be appended to the server name with an underscore character; otherwise, the ICP board number is assumed to be 0. The maximum length is ten characters.

dtime This is a pointer to a descriptor. It is a character string containing the timeout value time in the following time string format:

VMS format:	`dddd hh:mm:ss`
UNIX format:	`hh:mm:ss`

fixed This is a pointer to an integer. It is a longword containing the record format indicator. If the indicator is true, the file is opened with fixed record format.

recl This is a pointer to an integer. It is a longword containing the record length. A length of up to 4096 bytes is supported.

binary This is a pointer to an integer. It is a longword containing the boolean flag. If the flag is true, received data records are *not* deblocked using record separator characters, but by `recl` length only. A non-zero value is treated as true.

forms This is a pointer to an integer. It is a longword indicating the type of forms control (`FORTRAN`, `ESCAPE`, or `NONE` (default), defined as 1, 2, and 0, respectively) in the `BSCTRAN.INC` include file.

display This is a pointer to an integer. It is a longword containing the boolean flag for state change display. This flag is used to conditionally display the transmission state changes to the system output device.

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success RCOK

Failure RCILPAR, RCOPERR, RCFWRERR, RCFCLERR
Failure codes from BSCDAS, BSCASL, BSCREAD
Unexpected ICP response failure codes (Table 5–3 on page 98)

3.5 BSCSAFE Procedure

Calling Sequence

```
ReturnCode = BSCSAFE (lnum, code)
```

Procedures Called

BSCRESET and BSCFWY

Functions

- If code is true, sends a Safe Store Acknowledgment to the ICP and waits for the response.
- If code is false, sends a Safe Store Negative Acknowledgment to the ICP (no response).

Arguments

lnum This is a pointer to an integer. It is a longword containing the link number.

code This is an integer flag. If the flag is true, sends a Safe Store Acknowledgment to the ICP and waits for response. If the flag is false, sends a Safe Store Negative Acknowledgment (no response). A non-zero value is treated as true.

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success RCOK

Failure RCREJECT, RCNOSAFE, RCTO, RCQ_WWR, RCQ_RDTO
Unexpected ICP response failure codes (Table 5–3 on page 98)

3.6 BSCSEND Procedure

Calling Sequence

```
ReturnCode = BSCSEND (fname, lnum, server, dtime, trans,
                      binary, card, noteof, display, dak)
```

Procedures Called

BSCDAS, BSCASL, BSCWRIT, BSCRESET, and BSCFWY

Functions

1. Configures the DAK node to `dak` and issues a Flush Queue command to the ICP.

2. Opens the file (`fname`).

3. If there is a change in the server name or ICP device, calls BSCASL to assign input and output channels.

4. Accumulates file records one at a time until the buffer contains enough to fill the write block without splitting records. Inserts appropriate record separator character (2780 or 3780) between records.

Exceptions

   a. If the `binary` flag is true, no record separators are inserted, and each record is sent separately to the ICP.

   b. If the `DEFAULT /RECORD` option is set for 2780 records, and the `trans` flag is true, each record is preceded by a two-byte count, then packed into the write block without splitting records.

5. Calls BSCWRIT to send the block to link (`lnum`) and get ACK from the ICP.

6. Continues sending blocks until error, timeout (`dtime`), or end-of-file is returned. If timeout occurs, issues a Flush Queue command to the ICP.

7. Closes the file.

8. Logs the transmission statistics.

Arguments

fname This is a pointer to a descriptor. It is a character string containing the file specification of the file to be transmitted. The maximum length is 255 characters.

lnum This is a pointer to an integer. It is a longword containing the link number.

server This is a pointer to a descriptor. It is a character string containing the server name. An ICP board number may be appended to the server name with an underscore character; otherwise, the ICP board number is assumed to be 0. The maximum length is ten characters.

dtime This is a pointer to a descriptor. It is a character string containing the timeout value in the following time string format:

VMS format:	`dddd hh:mm:ss`
UNIX format:	`hh:mm:ss`

trans This is a pointer to an integer. It is a longword containing the transparent mode indicator. If the flag is true, the file is transmitted in transparent mode. A non-zero value is treated as true.

binary This is a pointer to an integer. It is a longword containing the binary indicator. If the flag is true, no record separators are inserted.

card This is a pointer to an integer. It is a longword containing the card image indicator. If the flag is true, each record is transmitted in 80-column card format. This option is not supported for transparent 2780 records.

noteof This is a pointer to an integer. It is a longword containing the noteof indicator. If the flag is true, the last block is sent as ETB rather than ETX.

display This is a pointer to an integer. It is a longword containing the boolean flag for state change display. This flag is used to conditionally log the transmission state changes to the system output device.

dak This is a longword containing the DAK node indicator. If the indicator is 1, the DAK node is configured *on*; otherwise it is configured *off*.

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success RCOK

Failure RCOPERR, RCFRDERR, RCTO, RCQ_WWR, RCQ_RDTO
Failure codes from BSCDAS, BSCASL, BSCWRIT
Unexpected ICP response failure codes (Table 5–3 on page 98)

3.7 BSCSET Procedure

Calling Sequence

```
ReturnCode = BSCSET (lnum, server, lconf, start)
```

Procedures Called

BSCDAS, BSCASL, BSCRESET, and BSCFWY

Functions

1. If there is a change in the server name or ICP device, calls BSCASL to assign input and output channels.

2. Sends the requested configuration(s) to the ICP for link (`lnum`) based on parameters `lconf` and `start`. Possibilities are:

   - Sets link configuration parameters
   - Starts (enables) ICP link (`lnum`)
   - Stops (disables) ICP link (`lnum`)

Arguments

lnum This is a pointer to an integer. It is a longword containing the link number.

server This is a pointer to a descriptor. It is a character string containing the server name. An ICP board number may be appended to the server name with an underscore character; otherwise, the ICP board number is assumed to be 0. The maximum length is ten characters.

lconf This is a pointer to a variable-length data structure. It contains the link configuration block or the autodial command string. The first word of the data structure contains the size in bytes of the data block

that follows. For link configuration, the data block consists of a variable number of two-word configuration parameter/value pairs. For autodial, the data block contains the NULL-terminated autodial command string, filled in as bytes. The data structure is defined by the following C typedef (refer to the `BSCTRAN.INC` include file):

```
typedef struct {          /* Link configuration          */
        short size ;      /* Size of conf_block in bytes */
        struct {
            short par ;   /* Parameter number            */
            short value ; /* Parameter value             */
        } conf_block[CNPARTOT];
} LCONF;
```

start This is a pointer to an integer. It is a longword containing a flag. If the flag is greater than zero, the ICP link is sent a start (enable) command. If the flag is set to `SFC_LOCKSTART` (52), the link is started and stays enabled until a stop (disable) command is sent, even if the disable is not sent by the current process (also see the caution in Chapter 6). If `lconf` has a link configuration data block (that is, the size field is nonzero), a configuration command is sent to the ICP prior to the start command. If the flag is zero, a stop (disable) command is sent.

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success RCOK

Warning RCSTARTED — The link was previously started with the lockstart option and has not been disabled.

Failure RCFAIL, RCTO, RCQ_WWR, RCQ_RDTO
Failure codes from BSCDAS, BSCASL
Unexpected ICP response failure codes (Table 5–3 on page 98)

3.8 BSCSHOW Procedure

Calling Sequence

```
ReturnCode = BSCSHOW (lnum, server, lconfig, clear_stats)
```

Procedures Called

BSCDAS, BSCASL, BSCRESET, and BSCFWY

Functions

1. If there is a change in the server name or ICP device, calls BSCASL to assign input and output channels.

2. If the `clear_stats` flag is 1, issues a Clear Statistics command (command code 4) for link (`lnum`) to the ICP, waits for the statistics report, and returns immediately. The report contains the statistics prior to clearing.

3. Issues requests for configuration, buffer, statistics, and status reports to ICP for link (`lnum`).

4. Gets reports and transfers configuration and status to data structure (`lconfig`).

Arguments

lnum This is a pointer to an integer. It is a longword containing the link number.

server This is a pointer to a descriptor. It is a character string containing the server name. An ICP board number may be appended to the server name with an underscore character; otherwise, the ICP board number is assumed to be 0. The maximum length is ten characters.

lconfig This is a pointer to a data structure to receive link configuration/status. The data structure is defined by the following C typedef (refer to the `BSCTRAN.INC` include file):

```
typedef struct                      /* LINK CONFIGURATION REPORT    */
{
           short lactive;           /* Local link active status     */
                                    /* Start buffer report          */
           short msgsize;           /* ICP message buffer size      */
           short msgfree;           /* # free ICP msg buffers       */
           short msgtot;            /* Total # ICP msg buffers      */
           short xmtsize;           /* Transmit message buffer size */
           short xmtfree;           /* # free ICP transmit buffers  */
           short xmttot;            /* Total # ICP transmit buffers */
           short linktot;           /* Total # of links             */
                                    /* Start statistics report      */
           short rcv_bcc;           /* BCC errors received          */
           short rcv_par;           /* Parity errors received       */
           short rcv_orun;          /* Receive overrun errors       */
           short rcv_buf;           /* Buffer errors (on receive)   */
           short snd_msg;           /* Data messages sent           */
           short rcv_msg;           /* Data messages received       */
           short snd_nak;           /* NAKs sent                    */
           short rcv_nak;           /* NAKs received                */
           short snd_buf;           /* Buffer errors (on send)      */
           short reserved;          /* Reserved                     */
           short snd_blks;          /* Transmission blocks sent     */
           short rcv_blks;          /* Transmission blocks received */
           short size;              /* Size of config report        */
           short config[CNPARTOT * 2]; /* Option/value pairs       */
} CONFRPT;
```

where `CNPARTOT` is equal to the total number of BSC configuration parameters.

clear_stats This is a longword containing the boolean flag. If the flag is 1, a Clear Statistics command (command code 4) is sent to the ICP for link (`lnum`).

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success	RCOK
Failure	RCTO, RCQ_WWR, RCQ_RDTO Failure codes from BSCDAS, BSCASL

3.9 Example Programs

The following BASIC example program calls BSCSEND:

```
10     %TITLE 'SEND'
20     external long FUNCTION BSCSEND( string BY DESC, &
                                       long BY REF, &
                                       string BY DESC, &
                                       string BY DESC, &
                                       long BY REF, &
                                       long BY REF, &
                                       long BY REF, &
                                       long BY REF, &
                                       long BY REF, &
                                       long BY VALUE )

30     declare long bsc_stat
40     declare long lnum
50     declare long trans, bin, card, noteof, display, dak

60     input "Enter TEXT file to send"; f$
70     input "Enter server name"; sv$
80     dt$   = "0 00:00:10"

90     input "Enter SEND link number"; lnum
100    trans = 0                         ! non-transparent
110    bin = 0                           ! include rec separators
120    card = 0                          ! no IBM card image
150    noteof = 0                        ! send ETX
160    display = 1                       ! display of state of changes
170    dak = 0                           ! dak node OFF

200    bsc_stat = BSCSEND( f$, lnum, sv$, dt$, trans, bin, &
                           card, noteof, display, dak )
210    print "BSCSEND return status = "; bsc_stat
       end
```

The following FORTRAN example program calls BSCRECV (the SYS$ASCTIM call applies to VMS only.):

```
program recv

        implicit none

        character*20 name_of_file /'filetest.dat'/
        integer*4 status

        integer*4 sys_stat, bsc_stat, BSCRECV
        integer*4 SYS$ASCTIM

        character*23 current_time

C  Argument list for BSCRECV
C_____________________________________________________________________

        character*20 fname
        byte bfname(20)! enable null termination
        integer*4 lnum, fixed
        character*10 server
        character*13 dtime
        integer*4 recl
        integer*4 binary
        integer*4 forms
        integer*4 display
        equivalence (fname,bfname)

C_____________________________________________________________________

1010    format ( '  ')
1012    format ( 10x, '----------------------1--------------------' )
1014    format ( 10x, a23)
1018    format ( 10x, a )

1       format( 10x, 'BSC return code = ' z8.8 )
2       format( 10x, 'Name of file:' )
3       format( 10x, a )

        character*60 recvfilemsg /'File received to port 3 spot'/
```

```
!-------------------------------------------------------------------
!      START OF CODE
!-------------------------------------------------------------------
!
!
C  Receive file from ICP
C-------------------------------------------------------------------

          fname = name_of_file
          bfname(13) = 0       ! null terminate string
          lnum = 3
          server = 'spot'
          dtime = '0000 00:05:00'
          recl = 80            ! record length
          binary = 0           ! include rec separators
          forms = 0            ! no forms control
          display = 1          ! display of state of changes

          bsc_stat = BSCRECV ( fname, lnum, server, dtime, fixed, recl,
                     1         binary, forms, display )

          sys_stat = SYS$ASCTIM( , current_time,, )
          write( unit=5, fmt=1010 )
          write( unit=5, fmt=1012 )
          write( unit=5, fmt=1014 ) current_time
          write( unit=5, fmt=1010 )
          write( unit=5, fmt=1018 ) recvfilemsg
          write( unit=5, fmt=1010 )
          write( unit=5, fmt=1 ) bsc_stat
          write( unit=5, fmt=2 )
          write( unit=5, fmt=3 ) name_of_file
          write( unit=5, fmt=1012 )

          end
```

Chapter

4 Record Transfer Interface (BSCRTI)

Note

In this document, the term "Freeway" can mean either a Freeway server or an embedded ICP. For the embedded ICP, also refer to the user's guide for your ICP and operating system (for example, the *ICP2432 User's Guide for OpenVMS Alpha (DLITE Interface)*).

The Record Transfer Interface consists of a set of procedure calls that are used by the BSC File Transfer Interface (BSCFTI) to send and receive records from the ICP. You may use these calls from your own application by linking your program with the modules `BSCRTI.OBJ`, `BSCFWY.OBJ`, and `ICPDLI.OBJ`.

The Record Transfer Interface requires more programming in the user application than the File Transfer Interface. The application has total responsibility for file access, record blocking, insertion of applicable record separators or count characters, and handling of errors or unexpected responses.

This chapter describes each BSCRTI procedure, and its associated parameters and return codes. See Table 5–2 on page 95 for a complete list of the BSC File Transfer program error codes.

4.1 BSCASL Procedure

This procedure must be executed before records can be read from or written to the ICP. It must be called on startup or when changing the link number, ICP number, or server name.

Applications calling BSCRTI procedures (such as BSCREAD or BSCWRIT) must call BSCDAS or BSCASL when changing the link number. BSCFTI procedures (BSCDIAL, BSCRECV, BSCSEND, BSCSET, and BSCSHOW) automatically monitor changes in link number.

Calling Sequence

```
ReturnCode = BSCASL (lnum, server)
```

Procedures Called

BSCRESET and BSCFWY

Functions

1. Assigns input and output channels to the server.
2. Configures ICP message buffer size to the value of the `DEFAULT /BUFSIZE` option.
3. Saves link (`lnum`) as a global value to be used by BSCRTI routines such as BSCREAD and BSCWRIT.
4. Calls BSCRESET.

Arguments

lnum This is a pointer to an integer. It is a longword containing the link number.

server This is a pointer to a descriptor. It is a character string containing the server name. An ICP board number may be appended to the server

name with an underscore character; otherwise, the ICP board number is assumed to be 0. The maximum length is ten characters.

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success	RCOK
Failure	RCALAS, RCERRIDNAME, RCQ_AS, RCQ_INIT

4.2 BSCDAS Procedure

This procedure should be executed before calling BSCASL to change the link number, ICP number, or server name.

Calling Sequence

```
ReturnCode = BSCDAS ()
```

Procedures Called

BSCFWY

Function

1. Deassigns current ICP input and output channels and cancels any outstanding I/O.

Arguments

None

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success RCOK

Failure RCNAS, RCQ_CAN

4.3 BSCMODEM Procedure

This procedure can be called only after the ICP has been assigned (see the BSCASL procedure, Section 4.1 on page 74).

This procedure issues an Autodial Start command (command code 38) or Modem Configuration command (command code 39) to the ICP and waits for a response.

Calling Sequence

```
ReturnCode = BSCMODEM (rdblock, string, dtime, code)
```

Procedures Called

BSCRESET and BSCFWY

Functions

1. Verifies I/O channel assignment and legal parameters.
2. Issues the appropriate command code to the ICP based on `code`.
3. Issues a read to the ICP and waits for completion, unexpected response, fatal error, or timeout (`dtime`).
4. Calls BSCRESET if a fatal error or timeout (`dtime`) occurs.

Arguments

rdblock This is a pointer to a data structure of type TRANBUF. It receives the response from the ICP. See Section 4.8 on page 91 or the `BSCTRAN.INC` include file for details of the data structure.

string This is a pointer to a descriptor. It is a character string containing the autodial start or modem configuration string.

dtime This is a pointer to a descriptor. It is a character string containing the timeout value in the following time string format:

VMS format:	`dddd hh:mm:ss`
UNIX format:	`hh:mm:ss`

code This is an integer. It is a longword containing the appropriate command code, either Autodial Start (command code 38) or Modem Configuration (command code 39).

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success RCOK

Failure RCNAS, RCILPAR, RCUNR, RCTO, RCQ_WWR, RCQ_RDTO

Caution

The RCUNR return code from BSCMODEM is not treated as a fatal error. See the Caution box in Section 4.7 on page 87 (BSCWRIT procedure) for an example of appropriate handling of the RCUNR return code. The RCUNR code is returned on any error received by the ICP from the modem.

4.4 BSCREAD Procedure

This procedure can be called only after the server, ICP, and link number have been assigned (see Section 4.1 on page 74).

Calling Sequence

```
ReturnCode = BSCREAD (rdblock, recbuf, recl, lastrec, fixed,
                      binary, forms, dtime)
```

Procedures Called

BSCRESET and BSCFWY

Functions

1. If the read block (`rdblock`) is empty, reads in another read block from the ICP. Determines if there is a Print or Punch component select sequence at the start of the first record in the read block and if so, deletes it.

2. Moves one record from the read block (`rdblock`) to the user buffer (`recbuf`). Deblocks records using record separators or `recl` characters, whichever is smaller. The record separators will not be returned to the caller.

   Exceptions:

   a. If the read block contains no record separators, it is treated as one record as long as there are fewer than or equal to `recl` characters.

   b. If the `binary` flag is true, ignores record separator characters and returns `recl` characters or entire read block contents, whichever is smaller.

   c. If the `fixed` flag is true, accumulates data of length (`recl`) before returning.

3. Sets last record flag (`lastrec`) when appropriate.

4. Calls BSCRESET if a fatal error or timeout (`dtime`) occurs.

Arguments

rdblock This is a pointer to a data structure of type TRANBUF. It receives the data block from the ICP. See Section 4.8 on page 91 or the `BSCTRAN.INC` include file for details of the data structure.

recbuf This is a pointer to a descriptor. It is a character string to receive the data record from the ICP. The maximum length is 4096 bytes. The descriptor length field should be cleared before calling BSCREAD.

Exception: After a RCUNR return from BSCREAD, the length should not be cleared, because it may contain a partial record length caused by the unexpected response.

recl This is a pointer to an integer. It is a longword containing the maximum record size to be returned. If the record in the current received block is longer than the maximum length, this record is split into `recl` size records (maximum = 4096).

lastrec This is a pointer to an integer. It is a longword boolean flag returned indicating whether the current record is the last record in the file. A non-zero value is treated as true. If this flag is true, BSCREAD received an ETX or EOT indication from the ICP. On input, a value of –1 signifies that the `rdblock` contains the first block of unexpected data, thus preventing BSCREAD from reading prematurely from the ICP. (See the example on page 82.)

fixed This is a pointer to an integer. It is a longword containing the boolean flag. If the flag is true, received data records are *not* deblocked using record separator characters, but by `recl` length only. `Fixed` takes precedence over `binary` for length.

binary This is a pointer to an integer. It is a longword containing the boolean flag. If the flag is true (and the fixed flag is false), received data records

are *not* deblocked using record separator characters, but by the length of the ICP read block or `recl`, whichever is smaller.

forms This is a pointer to an integer. It is a longword indicating the type of forms control, `FORTRAN`, `ESCAPE`, or `NONE` (default), defined as 1, 2, and 0, respectively, in the `BSCTRAN.INC` include file. If the input value is `NONE`, and `FORTRAN` or `ESCAPE` forms control characters are encountered at the start of the first record, the output value will be changed to either `FORTRAN` or `ESCAPE`. The forms control characters will remain at the start of the record to be interpreted by the calling routine.

dtime This is a pointer to a descriptor. It is a character string containing the timeout value in the following time string format:

VMS format:	`dddd hh:mm:ss`
UNIX format:	`hh:mm:ss`

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success RCOK

Failure RCNAS, RCOVRSZ, RCTO, RCUNR, RCQ_RDTO

Caution

The RCUNR return code from BSCREAD is treated as a non-fatal error. User applications that call this procedure should handle the unexpected response during record processing or call BSCRESET.

The C code fragment shown below illustrates the proper use of BSCREAD after an unexpected data response is received while calling BSCWRIT. See also the example following the BSCWRIT procedure in Section 4.7 on page 87. Note that the `rdblock` must be common between BSCWRIT and BSCREAD.

```
/* Use a common rdblock for BSCWRIT and BSCREAD                    */
   TRANBUF rdblock;          /* rdblock for ICP responses          */
             .
   struct dsc$descriptor_s rd_recbuf; /* Read record buffer        */
             .
   /* Open the receive file                                        */
             .
   lastrec = -1;  /* Tell BSCREAD that the global rdblock already  */
                  /* contains the first block of unexpected data   */

   do             /* Receive one record at a time until end-of-file */
   {
     rd_recbuf.dsc$w_length = 0;   /* Clear length field          */
     do           /* Loop until get a good record or error        */
     {
       retcode = BSCREAD (&rdblock, &rd_recbuf, &recl, &lastrec,
                          &fixed, &binary, &forms, &dtime);
       if (retcode == RCUNR)
       {
         /* Handle the unexpected response contained in the rdblock: */
         /* For example, you could ignore reports and abort on     */
         /* fatal errors.                                          */
       }
       else if (retcode != RCOK)
       {
          /* Abort program on fatal errors other than RCUNR       */
          /* (such as timeout of 'dtime' (RCTO error).            */
       }
      } while (retcode == RCUNR);
```

```
   /* Write the record in rd_recbuf to the receive file.        */
} while (!lastrec);       /* lastrec flag set by BSCREAD        */
                .
/* After the last record is received, close the file.           */
```

4.5 BSCRESET Procedure

This procedure is called when any error occurs that prevents the file transfer operation from completing normally.

Calling Sequence

```
BSCRESET ()
```

Procedures Called

BSCFWY

Functions

1. Cancels outstanding I/O.
2. Initializes all flags and counters used by BSCREAD and BSCWRIT.

Arguments

None

4.6 BSCTRACE Procedure

This procedure can be called only after the ICP has been assigned (see the BSCASL procedure, Section 4.1 on page 74).

This procedure is not used by BSCTRAN, but can be called by any user application linked with `BSCRTI.OBJ`.

Calling Sequence

```
ReturnCode = BSCTRACE (rdblock, dtime, start, stop, data)
```

Procedures Called

BSCRESET and BSCFWY

Functions

1. Verifies I/O channel assignment and legal parameters.
2. Issues writes to the ICP according to value of parameters `start`, `stop`, and `data`.
3. Issues appropriate reads to the ICP and waits for completion, fatal error, or timeout (`dtime`).
4. Calls BSCRESET if a fatal error or timeout (`dtime`) occurs.

Arguments

rdblock This is a pointer to a data structure of type TRANBUF. It receives the response from the ICP. See Section 4.8 on page 91 or the `BSCTRAN.INC` include file for details of the data structure.

dtime This is a pointer to a descriptor. It is a character string containing the timeout value in the following time string format:

VMS format:	`dddd hh:mm:ss`
UNIX format:	`hh:mm:ss`

start This is an integer. It is a longword containing the boolean flag. If the flag is true, BSCTRACE issues a start trace command to the ICP and waits for a start trace acknowledge.

stop This is an integer. It is a longword containing the boolean flag. If the flag is true, BSCTRACE issues a stop trace command to the ICP and waits for a stop trace acknowledge.

data This is an integer. It is a longword containing the boolean flag. If the flag is true, BSCTRACE issues a read to the ICP and waits for a trace data block or timeout (`dtime`).

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success RCOK

Failure RCNAS, RCILPAR, RCREJECT, RCTO, RCQ_WWR, RCQ_RDTO

4.7 BSCWRIT Procedure

This procedure can be called only after the ICP has been assigned (see the BSCASL procedure, Section 4.1 on page 74).

The caller has total responsibility for insertion of record separators or transparent 2780 count characters, and ensuring that the total ICP writeblock size does not exceed the value of the `DEFAULT /BUFSIZE` option (default is 1024).

To increase efficiency, BSCWRIT uses a write rotate window of two, allowing a maximum of two outstanding data acks at any one time. This significantly increases throughput compared to waiting for the data acknowledgment after every write to the ICP. To take advantage of this feature, applications calling BSCWRIT should use `action` parameters 1, 3, or 8 for the ETB blocks.

Calling Sequence

```
ReturnCode = BSCWRIT (rdblock, recbuf, action, dtime, dak)
```

Procedures Called

BSCRESET and BSCFWY

Functions

1. Transfers record in buffer (`recbuf`) to the ICP writeblock. BSCWRIT does not add record separator or count characters.

2. Based on `action`, sends the appropriate type of block to the ICP and waits for ACK if appropriate.

3. Calls BSCRESET after handling the ETX block or if a fatal error or timeout (`dtime`) occurs.

Arguments

rdblock This is a pointer to a data structure of type TRANBUF. It receives the data acknowledgment block from the ICP. See Section 4.8 on page 91 or the `BSCTRAN.INC` include file for details of the data structure.

recbuf This is a pointer to a descriptor. It is a character string containing the record to be sent. The maximum length is 4096 bytes. The record must contain any appropriate record separator or count characters.

action This is a pointer to an integer. It is a longword containing the action indicator. Using this parameter, the caller has total control over when the write block is sent to the ICP. BSCWRIT does not insert any record separator characters into the write block. Except for the BSC protocol characters (which are transparent to the user), the caller has total control of the block format.

0 = Move record to write block only and return immediately to caller. The maximum capacity of write block is defined by the `DEFAULT/BUFSIZE` option.

1[a] = Move record and send block as ETB.

2 = Move record, send block as ETX, and wait for ACK.

3[a] = Move record and send block as transparent ETB.

4 = Move record, send block as transparent ETX, and wait for ACK.

5[b] = Move record, send block as ETB, and wait for ACK.

7[b] = Move record, send block as transparent ETB, and wait for ACK.

8[a] = Move record and send block as transparent 2780 ETB.

10 = Move record, send block as transparent 2780 ETX, and wait for ACK.

12[b] = Move record, send block as transparent 2780 ETB, and wait for ACK.

[a] Actions 1, 3, and 8 are used for ETB blocks to enable BSCWRIT to use the write rotate window of two.

[b] Actions 5, 7, and 12 are for the last record (or block) of a file when using the SEND /NOTEOF option to concatenate files.

dtime This is a pointer to a descriptor. It is a character string containing the timeout value in the following time string format:

VMS format:	`dddd hh:mm:ss`
UNIX format:	`hh:mm:ss`

dak This is a longword containing the DAK node indicator. If the indicator is 1, data acknowledgments are read from the DAK node; otherwise, they are read from the link's read node.

Return Codes

See Table 5–2 on page 95 for full descriptions of the error return codes.

Success RCOK

Failure RCNAS, RCILPAR, RCOVRSZ, RCUNR, RCTO, RCQ_WWR, RCQ_RDTO

Caution

User applications that receive the RCUNR (non-fatal) return from BSCWRIT can cancel the SEND process by calling BSCRESET and issuing a Flush Queue command (command code 37).

Alternatively, the application can handle the unexpected response contained in the `rdblock` parameter. Then, to complete the interrupted SEND process, the same BSCWRIT call should be repeated until the return code is RCOK. This ensures that the expected data acknowledgment is received.

The C code fragment shown below illustrates the proper use of BSCWRIT to handle unexpected responses from the ICP while sending a file.

```
/* Use common rdblock for BSCWRIT and BSCREAD                          */
TRANBUF rdblock;              /* rdblock for ICP responses             */
          .
struct dsc$descriptor_s wr_recbuf; /* Write record buffer              */
          .
/*********************************************************************/
/* Call BSCASL to assign I/O channel.                                  */
/* Open the SEND file and get one record at a time.                    */
/* Build the write record buffer:  You can call BSCWRIT for each record*/
/* (with record separator appended), or concatenate several records    */
/* (separated by record separators).                                   */
/*********************************************************************/
          .
do              /* Loop to handle unexpected responses               */
{
  /* Write the current record buffer to the ICP                        */
  retcode = BSCWRIT (&rdblock, &wr_recbuf, &action, &dtime, FALSE);
  if (retcode == RCUNR)
  {
    /* Handle the unexpected response contained in the rdblock: for */
    /* example, you could ignore reports, abort on fatal errors,    */
    /* and write                                                    */
    /* unexpected data to a file by calling BSCREAD (see            */
    /* BSCREAD example).                                            */
  }
   else if (retcode != RCOK)
   {
   /* Abort program on fatal errors other than RCUNR                 */
   /* (such as timeout of 'dtime' (RCTO error).                      */
   }
} while (retcode == RCUNR);
          .
/* After the last record has been sent, close the file.              */
```

4.8 ICP Read Block

The ICP Read Block is a data structure used by the BSC File Transfer program to receive data or control information from the ICP. It is also used for error handling based on the value of the error status byte of the ICP control header. The entire data structure is described by the following C typedef (refer to the BSCTRAN.INC include file):

```
typedef struct
{
  struct                 /* CONTROL HEADER structure          */
  {
    short link;          /* Link number                       */
    char subfc;          /* Command code                      */
    char err_status;     /* Error status                      */
    short size;          /* Data size (in bytes)              */
    char cu;             /* Control Unit (N/A 3780)           */
    char du;             /* Device Unit (N/A 3780)            */
  } header;
    union                /* DATA ARRAY                        */
  {
    char bytes[MAXTRAN]; /* Access as characters (8 bits)     */
    short words[];       /* Access as short integers (16 bits) */
    long dwords[];       /* Access as long integers (32 bits)  */
  } data;
} TRANBUF;
```

MAXTRAN is defined as 4096.

Caution

The BSCTRAN program uses only one Read Block structure. It is used to receive data acknowledgments from the ICP during SEND operations and to receive data from the ICP during RECEIVE operations. The use of one common Read Block enables BSCTRAN to handle unexpected responses during either SEND or RECEIVE. User applications that call BSCWRIT and BSCREAD and wish to handle unexpected responses should also use a common Read Block structure. Refer to examples in Section 4.4 on page 79 and Section 4.7 on page 87.

Chapter

5 Error Handling

The BSC File Transfer program handles the following three categories of errors:

- Errors returned by the various levels of the BSC File Transfer program. These are described in Section 5.1.
- Errors returned by the ICP, described in Section 5.2.
- Unexpected responses from the ICP, described in Section 5.3.

Error return codes are defined in the `BSCTRAN.INC` include file.

5.1 BSCTRAN Program Warnings and Errors

Warning messages output by the BSCTRAN program during unexpected situations are listed in Table 5–1. BSCTRAN program error codes appear in Table 5–2 along with their probable causes. Chapter 3 and Chapter 4 include the applicable error codes that are used as return codes for each of the procedures.

Table 5–1: Warnings Output by BSC File Transfer Program

Warning Message	Specified Area
Link is stopped	Link number (after a fatal error and abort)
Link already started	Link number
Receiving carriage control	`/ESCAPE` or `/FORTRAN`
Received empty file	Filename
File deleted	Filename
Partial file	Filename
Unexpected file	Default filename

Table 5–2: Error Return Codes Used by BSC File Transfer Program

(defined in the BSCTRAN.INC include file)

Mnemonic	Value (hex)	Meaning	Probable Cause
RCOK	0001	No error	The only success code
RCSTARTED	0002	Link already started (enabled)	Previous `SET /LOCKSTART` command
RCERRIDNAME	8012	Invalid ICP device name	BSCTRAN command line error
RCFAIL[a]	8022	Link not started	No DSR signal
RCNAS	8032	ICP not previously assigned	User application error
RCALAS	80A2	ICP already assigned	User application error
RCILPAR	8042	Illegal parameter value	User application error
RCOVRSZ	8052	Record exceeds maximum size	BSCWRIT: Record > `DEFAULT /BUFSIZE` BSCREAD: Trans 2780 rec > `recl` param
RCTO[b]	8062	BSCTRAN timeout	SEND or RECEIVE `dtime` expired
RCUNR[c]	8072	Received unexpected response	Expect data acks on SEND, data on RECEIVE
RCOPERR[d]	8082	File open error	See footnote d
RCFRDERR[d]	8092	File read error	See footnote d
RCFCLERR[d]	8142	File close error	See footnote d
RCFWRERR[d]	80B2	File write error	See footnote d
RCILOPT	80D2	Illegal option	BSCTRAN command line error
RCNOLNK	80E2	No link number specification	BSCTRAN command line error
RCNOFNAME	80F2	No filename specification	BSCTRAN command line error
RCIOPVAL	8102	Illegal value for option	BSCTRAN command line error
RCREJECT	8112	Transmission rejected	Refer to Table 5–3 on page 98
RCNOSAFE	8152	Safe store abort by remote	Remote timed out waiting for ACK
RCQ_AS	8172	BSCFWY/icpas error	VMS `SYS$ASSIGN` error
RCQ_INIT	8182	BSCFWY/icpinit error	VMS event flag error

Table 5–2: Error Return Codes Used by BSC File Transfer Program *(Cont'd)*

(defined in the BSCTRAN.INC include file)

Mnemonic	Value (hex)	Meaning	Probable Cause
RCQ_WWR	8192	BSCFWY/icpwwr error	Write buffer size > ICP buffersize
RCQ_CAN	81A2	BSCFWY/icpcanqio/icpcnq error	VMS SYS$CANCEL or SYS$DASSGN error
RCQ_RDTO	81B2	BSCFWY/icprdtimo error	Read buffer size too small
RCAUTODIAL[b]	8162	ICP BE_AUTODIAL error	Refer to Table 5–3 on page 98
RCEOT[b]	81C2	ICP BE_EOT_ABORT error	Refer to Table 5–3 on page 98
RCXRETRY[b]	81D2	ICP BE_XRETRY error	Refer to Table 5–3 on page 98
RCNOBUFS[b]	81E2	ICP BE_NOBUFS error	Refer to Table 5–3 on page 98
RCDCDTO[b]	81F2	ICP BE_DCD_TO error	Refer to Table 5–3 on page 98
RCRVI[b]	8202	ICP BE_RVI_ABORT error	Refer to Table 5–3 on page 98
RCDLEEOT[b]	8212	ICP SFC_DLEEOT subfn code	The ICP received a DLEEOT

[a] Refer to the SET /START command in Section 2.4 on page 31.

[b] If this error occurs during SEND or RECEIVE and the DEFAULT /ABORT option is set, BSCFTI will abort by calling LIB$STOP.

[c] RCUNR is treated as a non-fatal error and is handled by the BSCFTI procedures so that the SEND or RECEIVE operation can continue to normal completion.

[d] For errors during file accesses, two VMS error messages (RMS error code and VMS system error code) are logged to SYS$OUTPUT and applicable log files.

5.2 BSC ICP Errors

Table 5–3 describes the possible errors returned from the ICP and how each is handled in SEND and RECEIVE modes. If the DEFAULT /ABORT option is set, then for all cases causing an abort of the BSC File Transfer program, the applicable link is stopped and an informational message is sent to the system output device advising the user to restart the link.

If the DEFAULT /NOABORT option is set, the user is returned to the BSCTRAN command prompt. User applications receive the BSCFTI return code indicated in Table 5–3.

Table 5–3: BSC ICP Error Return Codes

(defined in the BSCTRAN.INC include file)

Mnemonic	Value (decimal)	Error Message/BSCTRAN Disposition	BSCFTI Return Code
BE_EOT_ABORT	3	Transmission aborted by EOT Send: Abort[a] Recv: Delete or save[b] partial file; Abort[a]	RCEOT
BE_XRETRY	5	Retry limited exceeded Send: Abort[a] Recv: Delete or save[b] partial file; Abort[a]	RCXRETRY
BE_NOBUFS	7	Insufficient ICP message buffers to start link BSCSET: Abort[a]	RCNOBUFS
BE_DCD_TO	11	DCD timeout Send: Abort[a] Recv: Delete or save[b] partial file; Abort[a]	RCDCDTO
BE_RVI_ABORT	13	Received RVI; transmission aborted Send: Abort[a] Recv: N/A	RCRVI
BE_BUSY[c]	17	Line busy (after BSCSEND Flush Queue) Send: Continue with SEND operation Recv: N/A	RCOK from BSCFTI RCUNR from BSCRTI
BE_USER_ABORT[c]	29	Transmission aborted by user Send: Continue with SEND operation Recv: Continue with RECEIVE operation	RCOK from BSCFTI RCUNR from BSCRTI
BE_AUTODIAL	31	Autodial failure Send: Abort[a] Recv: N/A	REAUTODIAL
BE_REJECT	-128	Transmission rejected Send: Continue with next operation Recv: Continue with next operation	RCREJECT

[a] Depending on current setting of DEFAULT /ABORT option.

[b] Depending on current setting of DEFAULT /DELETE option.

[c] BSCSEND or BSCRECV will ignore this (unlikely) error in an attempt to complete the SEND or RECEIVE operation. An error message is logged to the system output device and applicable log files. User applications that need to catch this error can check the error field in the BSC header after the RCUNR return code from BSCREAD or BSCWRIT.

5.3 Unexpected Responses from the ICP

Unexpected responses from the ICP can be in the form of data, status, or control information due to previous or interfering activity on the line. They can also be caused by performing BSC File Transfer commands in an improper order such as executing a SEND or SET/STOP command when data has already been received from the remote site and is awaiting a RECEIVE command.

Status information (for examples, configuration reports) will be ignored. Any unexpected data will be copied into a default file named DEFAULT.DAT (or user's default file name defined with the DEFAULT /FILE command) and an informative message sent to the system output device.

Chapter

6 Cautions

The following is a list of cautions that users of the BSC File Transfer program should heed.

- The default size is 1024 bytes for the BSCTRAN write buffer and the ICP message buffers, but the value can be increased up to 4096 by means of the `DEFAULT /BUFSIZE=n` command. The BSCTRAN write buffer size must always be less than or equal to the ICP message buffer size to avoid buffer errors from Freeway. Refer to Section 2.5 on page 33 for details of the `DEFAULT /BUFSIZE` command. See the *BSC Programmer's Guide* for more information on ICP message buffers.

- BSCTRAN posts all reads to the ICP with a read buffer size of 4096 plus 8 bytes for the control header (regardless of the ICP buffer size). Therefore, your VMS SYSGEN MAXBUF parameter must be at least 5004, or you will get an "exceeded quota" error. Consult your system manager if you encounter this problem.

- For some configuration parameters, a link must be disabled prior to issuing a `SET /PARn=d` command. Refer to the *BSC Programmer's Guide*.

- The `SET /START` command must be issued before sending or receiving files on a link.

- When sending or receiving transparent 3780 data, use the `/BINARY` option to prevent the addition or deletion of record separator characters. For transparent 2780 data, use the `/BINARY` option only on receive.

- When transmitting 3780 executable-image files between VMS or UNIX sites, use the commands `SEND /TRANS /BINARY` and `RECEIVE /FIXED /BINARY`.

- When transmitting 3780 object-image files between VMS or UNIX sites, use the commands `SEND /TRANS /BINARY` and `RECEIVE /BINARY`. Be sure the transmission block size (configuration option 8) is larger than the largest record.

- When sending and/or receiving transparent data, both the local and remote links must be configured to EBCDIC character set translation (the default) or ASCII with no parity.

- Refer to the *BSC Programmer's Guide* regarding the relationship between the ICP data translation configuration option and various types of data.

- In general, it is a good idea to set the ICP transmission block size greater than or equal to the maximum record size + 2. This avoids splitting records and also increases throughput.

- Be aware of the implications of using the `SET /LOCKSTART` command. After a link is enabled with `/LOCKSTART`, it remains enabled (even if BSCTRAN terminates) until the link is disabled with a `SET /STOP` command. A link that remains enabled after termination of a BSCTRAN process (due to a `LOCKSTART`) continues receiving messages into its data queues until it either runs out of buffers, or until a new BSCTRAN process references (attaches) the same link. As soon as an attach is made, data is sent to BSCTRAN. Therefore, in order to receive data accumulated on a link, the first command must be `RECEIVE`. Any other command discards received data that is not relevant to the command. For example, a `SHOW` command discards any data packets that are not responses to its report requests.

Chapter

7 Example BSCTRAN Usage

Note

In this document, the term "Freeway" can mean either a Freeway server or an embedded ICP. For the embedded ICP, also refer to the user's guide for your ICP and operating system (for example, the *ICP2432 User's Guide for OpenVMS Alpha (DLITE Interface)*).

This chapter shows an example of how BSCTRAN is used. The BSCTRAN program requires that you use the supplied DLI and TSI configuration files which are named `dlitrancfg` and `tsitrancfg` respectively. If you modify these files, you must regenerate their binary versions (`dlitrancfg.bin` and `tsitrancfg.bin`). Do not change the names of these files. For more information on `dlitrancfg` and `tsitrancfg`, refer to Section 2.1 on page 20. For more information about DLI and TSI configuration files in general, see the *Freeway Data Link Interface Reference Guide* and the *Freeway Transport Subsystem Interface Reference Guide*. For the embedded ICP, also refer to the user's guide for your ICP and operating system (for example, the *ICP2432 User's Guide for OpenVMS Alpha (DLITE Interface)*).

7.1 DLI Configuration File Example

Figure 7–1 is an example of a DLI configuration file that defines several sessions.

```
//--------------------------------------------------------------------//
// DLI configuration file for BSCTRAN                                 //
//                                                                    //
// This file defines the connections between the BSCTRAN program      //
// and individual ICP ports on the Freeway server(s).  In this        //
// file, a main section is defined along with a section for each      //
// ICP port. The names of each section correspond to the names        //
// used in the BSCTRAN program to identify an individual Freeway,     //
// ICP board, and port.  For example, the BSCTRAN command:            //
//                                                                    //
//                     SHOW 2 FWYO_1                                  //
//                                                                    //
// would use connection name "FWYOB1L2" which would access port 2     //
// on ICP board 1 on Freeway O.  The defaults of most of the main     //
// section parameters are used, so the parameters do not show up      //
// in this file. See the DLI reference manual for a definition of     //
// all DLI configuration parameters.                                  //
//                                                                    //
// If your system has more Freeway servers or ICP boards than are     //
// defined in this file, duplicate a section of this file, paste      //
// it to the end, then edit the pasted connection names as per the    //
// example above and recompile the configuration file.                //
//                                                                    //
//--------------------------------------------------------------------//

//--------------------------------------------------------------------//
//                                                                    //
// "main" section.  If not defined defaults are used. If present      //
// the main section must be the very first section of the DLI         //
// configuration file.                                                //
//                                                                    //
//--------------------------------------------------------------------//
main
{
    AsyncIO = "yes";                 // Asynchronous mode            //
    TSICfgName = "tsitrancfg.bin";   // TSI configuration file name  //
                          // or TSICfgName = "." (if using DLITE)    //
    loglev = 3;
    tracelev = 3;
    traceSize = 64000;
    traceName = "dlibsc.trc";
// The following two parameters are for DLITE only:                   //
   MaxBuffers = 1024;
   MaxBufSize = 1200;
}
```

Figure 7–1: DLI Configuration File for BSCTRAN

```
//--------------------------------------------------------------------//
// Define 16 port names for Freeway 0, ICP board 0.                   //
// The transport connection name "bsc0" is used for all ports on      //
// Freeway 0.  In order to complete the connection you must replace   //
// the dummy server name "Freeway0" in the tsi configuration file     //
// with the name of your Freeway server.                              //
//--------------------------------------------------------------------//
FWY0BOL0
{
    Protocol = "RAW";                // RAW session type              //
    Transport = "bsc0";              // Transport connection name     //
                                     // defined in TSI configuration//
    BoardNo = 0;                     // ICP board number -- based 0 //
    PortNo = 0;                      // Port number                   //
    Async10 = "yes";
    Always0io = "yes";
    Timeout = 0;                     // Zero means never timeout      //
    loglev = 3;
    tracelev = 3;
}
FWY0BOL1
{
    Protocol = "RAW";                // RAW session type              //
    Transport = "bsc0";              // Transport connection name     //
                                     // defined in TSI configuration//
    BoardNo = 0;                     // ICP board number -- based 0 //
    PortNo = 1;                      // Port number.                  //
    Async10 = "yes";
    Always0io = "yes";
    Timeout = 0;                     // Zero means never timeout      //
    loglev = 3;
    tracelev = 3;
}
FWY0BOL2
{
    Protocol = "RAW";                // RAW session type              //
    Transport = "bsc0";              // Transport connection name     //
                                     // defined in TSI configuration//
    BoardNo = 0;                     // ICP board number -- based 0 //
    PortNo = 2;                      // Port number.                  //
    Async10 = "yes";
    Always0io = "yes";
    Timeout = 0;                     // Zero means never timeout      //
}
```

Figure 7–1: DLI Configuration File for BSCTRAN *(Cont'd)*

```
FWYOBOL3
{
    Protocol = "RAW";                // RAW session type            //
    Transport = "bsc0";              // Transport connection name   //
                                     // defined in TSI configuration//
    BoardNo = 0;                     // ICP board number -- based 0 //
    PortNo = 3;                      // Port number.                //
    Async10 = "yes";
    AlwaysQio = "yes";
    Timeout = 0;                     // Zero means never timeout    //
}
.
.
.

//-------------------------------------------------------------------//
// Define 16 port names for Freeway 1, ICP board 0.                  //
// The transport connection name "bsc1" is used for all ports on     //
// Freeway 1.  In order to complete the connection you must replace  //
// the dummy server name "Freeway1" in the tsi configuration file    //
// with the name of your Freeway server.                             //
//-------------------------------------------------------------------//
FWY1BOL0
{
    Protocol = "RAW";                // RAW session type            //
    Transport = "bsc1";              // Transport connection name   //
                                     // defined in TSI configuration//
    BoardNo = 0;                     // ICP board number -- based 0 //
    PortNo = 0;                      // Port number                 //
    Async10 = "yes";
    AlwaysQio = "yes";
    Timeout = 0;                     // Zero means never timeout    //
    loglev = 3;
    tracelev = 3;
}
.
.
.
```

Figure 7–1: DLI Configuration File for BSCTRAN *(Cont'd)*

7.2 TSI Configuration File Example (Freeway Server Only)

Figure 7–2 is an example of a TSI configuration file that defines two connections. The connections specify their Freeway servers to be `freeway0` and `freeway1`, respectively.

```
//---------------------------------------------------------------------------//
// TSI configuration file for BSCTRAN                                        //
//                                                                           //
// This file defines TSI parameters for the BSCTRAN file transfer program.   //
// Each Freeway server is defined as a separate section.  The defaults of    //
// most of the main section parameters are used, so the parameters do not    //
// show up in this file. See the TSI reference manual for a definition of    //
// all TSI configuration parameters.                                         //
//                                                                           //
//---------------------------------------------------------------------------//

//---------------------------------------------------------------------------//
//                                                                           //
// "main" section: if defined, must be the very first section.               //
//                                                                           //
//---------------------------------------------------------------------------//
main
{
    asyncio = "yes";
    maxbufsize = 1200;                  // Always add 200 bytes to desired   //
                                        // data size for buffer overhead     //
    maxbuffers = 256;
    loglev = 7;
    tracelev = 3;
    traceSize = 64000;
    traceName = "tsibsc.trc";
}
```

Figure 7–2: TSI Configuration File for BSCTRAN

```
//--------------------------------------------------------------------------//
// bsc0: TCP socket interface connection for BSCTRAN on Freeway 0.          //
//                                                                          //
// Change the Freeway server name below to the name of your Freeway.        //
//--------------------------------------------------------------------------//
bsc0
{
    asyncio = "yes";
    transport = "tcp-socket";
    server = "freeway0";                // Your Freeway server name here     //
    wellknownport = 0x'2010';
    timeout = 0;                        // Zero means never timeout          //
    tracelev = 3;
    loglev = 7;
}

//--------------------------------------------------------------------------//
// bsc1: TCP socket interface connection for BSCTRAN on Freeway 1.          //
//                                                                          //
// Change the Freeway server name below to the name of your Freeway.        //
//--------------------------------------------------------------------------//
bsc1
{
    asyncio = "yes";
    transport = "tcp-socket";
    server = "freeway1";                // Your Freeway server name here     //
    wellknownport = 0x'2010';
    timeout = 0;                        // Zero means never timeout          //
    tracelev = 3;
    loglev = 7;
}
```

Figure 7–2: TSI Configuration File for BSCTRAN *(Cont'd)*

7.3 Command Procedure Example

The VMS DCL command procedure shown in Figure 7–3 demonstrates most of the commands and options of the BSCTRAN program. The command procedure uses DLI sessions `FWYOBOL2` and `FWYOBOL3`. The connection is `bsc0`, and the Freeway server name is `freeway0`. The demonstration requires Freeway server `freeway0`, ICP 0, links 2 and 3 to be connected with a loopback cable. As a rule, Freeway servers are configured at Simpact for external clocks, because in a production environment the clocking is provided by modems (that is, the clocking is external).

Note

A UNIX command file would omit the beginning and final lines (those lines starting with a "$").

This procedure assumes that the BSC 3780 software is already downloaded to the ICP. In an actual situation using two remote sites, this command procedure would be split into two parts, one to control each site.

Caution

All files sent in a loopback test, such as the Figure 7–3 example, must be small enough to be entirely contained within ICP memory, or the ICP will hang during the send operation. The files used in the SEND commands of Figure 7–3 are not provided with the BSCTRAN product. If you wish to use a similar procedure to test BSCTRAN, you can use any moderately sized text, object, and executable files for the SEND commands.

Note

The procedure in Figure 7–3 starts link 2, then link 3. A diagnostic message will be reported indicating that link 2 has not started. You can ignore this message because when link 3 starts, link 2 will detect link 3's carrier signal through the loopback cable. This will cause link 2 to start itself.

```
$  ! Set up to catch fatal errors if BSCTRAN has to abort.
$  ON ERROR THEN GOTO END
$  ! Set to echo the BSCTRAN commands to SYS$OUTPUT.
$  SET VERIFY=IMAGE
$  RUN BSCTRAN
HELP                    ! Display the Help menu (comments are allowed)
!
!                       You only have to specify the server name once.
SET 2 fwy0 /PAR2=0      ! Set  EXTERNAL clock for loopback
SET 2 /START            ! Start Link 2
SET 3 /PAR2=1 /START    !    or can configure and start together !
SHOW 2                  ! Show link configuration
SHOW 3
!
SEND TXT.TEST 2         ! Send and receive a text file
RECEIVE TEST.TXT 3
!
SEND EXE.TEST 2 /TRANS /BIN   ! Send and receive an executable file
RECEIVE TEST.EXE 3 /FIX /BIN  ! /RECL will be default of 512
!
DEFAULT /FILE=EXE.DAT /FIX /BIN /RECL=512
                              ! User-defined default file
SHOW /DEFAULT                 ! Show global options
SEND EXE.TEST 2 /TRANS /BIN   ! Send an executable file
SHOW 3                        ! Receive in EXE.DAT default file
!
SEND TXT1.TEST 2 /NOTEOF      ! Concatenate 3 files
SEND TXT2.TEST 2 /NOTEOF      ! #2 of 3
SEND TXT3.TEST 2              ! #3 of 3 (don't use /NOTEOF option)
RECEIVE TEST.CONCAT 3
!
SEND TXT.TEST 2 /CARD         ! Send and receive a card-image file
RECEIVE TEST.CARD 3
!
SEND ESCFOR.TEST 2            ! Send a Fortran forms-control file
RECEIVE TEST.ESCFOR 3 /FOR    ! (with escape sequence)
!
SEND ESCFOR.TEST 2            ! Send the same file again
RECEIVE TEST.ESCFOR 3         ! It will default even without /FORTRAN
!
SEND FOR.TEST 2         ! Send a Fortran type without escape sequence
RECEIVE TEST.FOR 3 /FOR ! Without ESC char, must specify /FOR
!
```

Figure 7–3: VMS DCL Command Procedure Example

```
SET 2 /STOP                  ! Disable to reconfigure link to send
SET 3 /STOP                  ! transparent data
!
SET 2 /PAR10=0/PAR19=0       ! Turn data translation and message
SET 3 /PAR10=0/PAR19=0       ! blocking off for transparent data
SET 2 /PAR8=514/START        ! Change block size to transmit full 512
SET 3 /PAR8=514/START        ! bytes (514 = 512 + STX + ETX)
SEND OBJ.TEST 2 /TRANS /BINARY     ! /BIN = don't insert RS chars
RECEIVE TEST_3780.OBJ 3 /BINARY    ! /BIN = ignore RS chars
!
DEFAULT /RECORD=2780         ! Set BSCTRAN to send 2780
SEND OBJ.TEST 2 /TRANS       ! No /BIN on send since transparent
                             ! 2780 uses count
RECEIVE TEST_2780.OBJ 3 /BINARY ! Always use /BIN to receive transparent
!
SET 2 /STOP                  ! Disable links
SET 3 /STOP
!
EXIT                         ! Exit the BSC File Transfer program
$ END:
```

Figure 7–3: VMS DCL Command Procedure Example *(Cont'd)*

Index

Customer Report Form

We are constantly improving our products. If you have suggestions or problems you would like to report regarding the hardware, software or documentation, please complete this form and mail it to Simpact at 9210 Sky Park Court, San Diego, CA 92123, or fax it to (619) 560-2838.

If you are reporting errors in the documentation, please enter the section and page number.

Your Name: _____

Company: _____

Address: _____

_____

_____

Phone Number: _____

Product: _____

Problem or
Suggestion: _____

_____

_____

_____

_____

_____

Simpact, Inc.
Customer Service
9210 Sky Park Court
San Diego, CA 92123