Freeway® Message Switch User's Guide

DC 900-1588D

Protogate, Inc.

12225 World Trade Drive
Suite R
San Diego, CA
92128
USA
Web: www.protogate.com
Email: sales@protogate.com
Voice: (858) 451-0865
Fax: (877) 473-0190

Freeway® Message Switch User's Guide: DC 900-1588D
by Protogate, Inc.

Published January 2001

This User's Guide covers the installation and configuration of the Message Switch, a Server Resident Application (SRA) which runs in a Freeway server.

This copy of the Message Switch User's Guide corresponds to version 3.0-1 of the Message Switch software. The latest version of this document is always available, in a variety of formats and compression options, from the Protogate World Wide Web server (http://www.protogate.com/support/manuals).

Table of Contents

List of Tables

List of Figures

Preface

1. Purpose of Document

This document describes the operation and configuration of Simpact's Message Switch software running in a Freeway communications server.

2. Intended Audience

This document should be read by programmers who are interfacing a client application to Freeway using the Message Switch software, and by users who want to configure a Freeway to switch data streams between different WAN (Wide-Area Network) lines, IP (Internet Protocol) addresses or types, and data formats.

3. Organization of Document

This document is organized into the following major sections:

Chapter 1

is an overview of the Message Switch, Freeway, and the Data Link Interface (DLI). This section includes a diagram of an example Message Switch configuration.

Chapter 2

describes Message Switch configuration and startup.

Appendix A

describes how to use the Message Switch `switch.cfg` file.

Appendix B

discusses other Message Switch configuration files that support the `switch.cfg` file described in Appendix A, such as the `bootcfg.sw` file, the `swdcfg` file, and the `swtcfg` file.

4. Protogate References

The following general product documentation list is to familiarize you with the available Protogate Freeway and embedded ICP products. The applicable product-specific reference documents are mentioned throughout each document (also refer to the "readme" file shipped with each product). Most documents are available on-line at Protogate's web site, www.protogate.com.

General Product Overview Documents

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 Documents

Freeway 500 Hardware Installation Guide	DC-900-2000
Freeway 1100/1150 Hardware Installation Guide	DC-900-1370
Freeway 1200/1300 Hardware Installation Guide	DC-900-1537
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 Electrical Interfaces (Addendum to DC-900-1501)	DC-900-1566
ICP2432 Hardware Installation Guide	DC-900-1502

Freeway Software Installation and Configuration Support Documents

Freeway Message Switch User Guide	DC-900-1588
Freeway Release Addendum: Client Platforms	DC-900-1555
Freeway User Guide	DC-900-1333
Freeway Loopback Test Procedures	DC-900-1533

Embedded ICP Software Installation and Programming Support Documents

ICP2432 User Guide for Digital UNIX	DC-900-1513
ICP2432 User Guide for OpenVMS Alpha	DC-900-1511
ICP2432 User Guide for OpenVMS Alpha (DLITE Interface)	DC-900-1516
ICP2432 User Guide for Solaris STREAMS	DC-900-1512
ICP2432 User Guide for Windows NT	DC-900-1510
ICP2432 User Guide for Windows NT (DLITE Interface)	DC-900-1514

Application Program Interface (API) Programming Support Documents

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 Documents

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

Toolkit Programming Support Documents

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

Protocol Support Documents

ADCCP NRM Programmer Guide	DC-900-1317
Asynchronous Wire Service (AWS) Programmer Guide	DC-900-1324
AUTODIN Programmer Guide	DC-908-1558
Bit-Stream Protocol Programmer Guide	DC-900-1574

BSC Programmer Guide	DC-900-1340
BSCDEMO User Guide	DC-900-1349
BSCTRAN Programmer Guide	DC-900-1406
DDCMP Programmer Guide	DC-900-1343
FMP Programmer Guide	DC-900-1339
Military/Government Protocols Programmer Guide	DC-900-1602
N/SP-STD-1200B Programmer Guide	DC-908-1359
SIO STD-1300 Programmer 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

5. Document Conventions

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

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 always uses the term "link."

6. Revision History

The revision history of the Freeway Message Switch User’s Guide, Protogate document DC 900-1588, is recorded below:

Table 1. Revision History

Revision	Release Date	Description
DC 900-1588A	August 1999	Original Release
DC 900-1588B	December 1999	Added descriptions of new TCP/IP listening and flow control capabilities. Updated throughout for latest bootcfg.sw and switch.cfg files. Added Message Switch diagram (Figure 1-2)
DC 900-1588C	September 2000	Rewritten in DocBook source format.

Revision	Release Date	Description
DC 900-1588D	January 2001	Updated Protogate’s address; minor formatting changes.

7. Customer Support

If you are having trouble with any Protogate product, call us at 1-858-451-0865 (U.S.) Monday through Friday between 8 a.m. and 5 p.m. Pacific time. You can also fax your questions to us at (858) 451-2865 or (877) 473-0190 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. Introduction

1.1. Message Switch Overview

The Message Switch is a Server-Resident Application (SRA) which runs in a Freeway server. It implements a switching function which transfers all data received by a set of client application sessions to one or more other sessions. The Message Switch software uses Simpact's Data Link Interface (DLI to open and manage Freeway client sessions to transmit and receive data.

The following is a quick overview of how to use the Message Switch software:

- Customize the provided configuration files to define the required client sessions for your Freeway environment, and to load and run the Message Switch.
- Boot the Freeway server. The Message Switch automatically configures itself, uses DLI to attach to one or more ICP ports, and begins transferring data. (ICP boards, or Intelligent Communications Processor boards, are the Protogate hardware product which implement various data communications protocols for sending and receiving data on a WAN link.)
- To communicate through the Message Switch, a client application can open a socket and send and receive data using the Freeway's IP address. Example programs are provided for testing purposes.

The following types of data can be exchanged through the Message Switch:

- Loopback
- Multicast (on Freeway platforms that support Multicast)
- Unicast
- WAN

The Message Switch offers an extremely flexible range of data-switching capabilities, all of which are available simply by modifying configuration files. For example, it is possible to use the Message Switch to translate data streams between different protocols, or between similar protocols with different characteristics (see Section A.3). Data streams can be replicated to multiple destinations (see Section A.4). Data streams from several sources can be combined into one (or more) destinations (see Section A.5). And the sources or destinations of any of these data streams can be WAN lines or IP unicast or multicast addresses, in any combination.

In addition, a Freeway which is running the Message Switch can be used by traditional DLI client application programs, in either Raw or Normal operation, and using either Simpact's DLI library or a simple socket interface (see Section A.6).

The Message Switch can greatly simplify the writing of client application programs which do not use Simpact's DLI library. For example, the Message Switch relieves the client application of the need to send and receive all the packets which are normally required to attach, bind, configure, and enable each data link. This is possible because the Message Switch does all the work necessary to bring up a link using data from its own configuration files (such as `swdcfg`).

After the Freeway and Message Switch are running, the client application program can simply open a socket and start sending and receiving data. For an example, see Section A.8 and the remarkably short program `freeway/client/test/switch/apinull.c`.

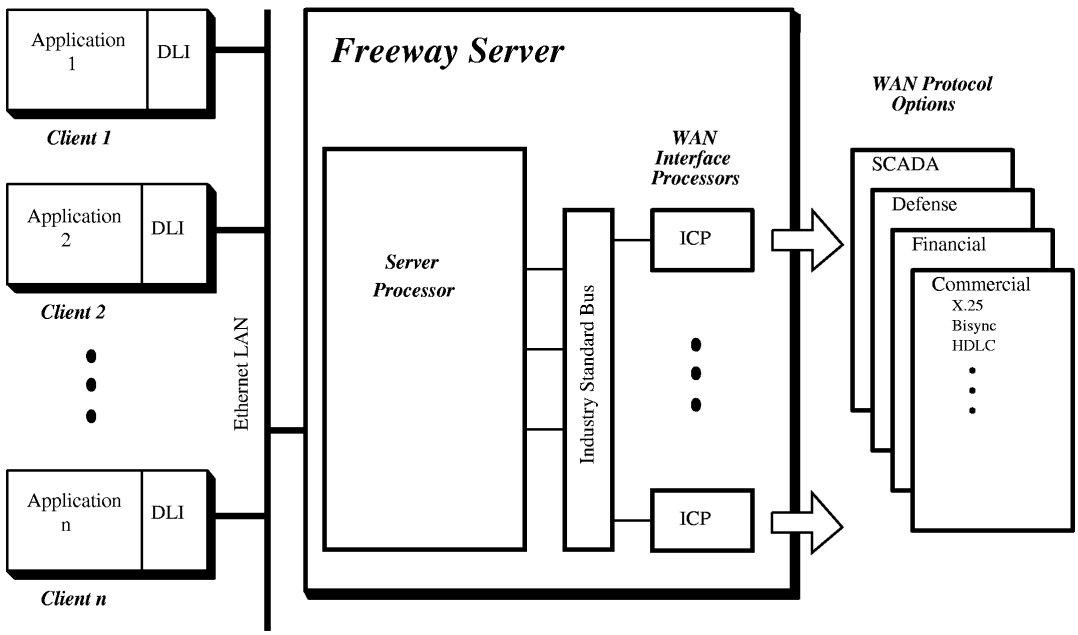
1.2. Freeway Server

Simpact provides a variety of user-programmable, wide-area network (WAN) connectivity solutions for real-time financial, defense, telecommunications, and process-control applications. Simpact software includes a wide variety of legacy and specialty protocols that run on Simpact's intelligent communication processor (ICP) boards residing in a Freeway communications server. The protocol software is independent of the hardware and operating system environment.

The Freeway server is a stand-alone box with pre-installed ICPs. It provides multiple data links and a variety of network services to LAN-based clients. Figure 1-1 shows a Freeway server configuration where the client application communicates with the Freeway ICPs using Simpact's data link interface (DLI). A variety of client operating systems are supported (UNIX, VMS, and Windows NT).

The Message Switch runs in the Freeway server processor (shown in Figure 1-1) and uses DLI to access WAN and IP links through ICP boards.

Figure 1-1. Freeway Configuration (With Message Switch)

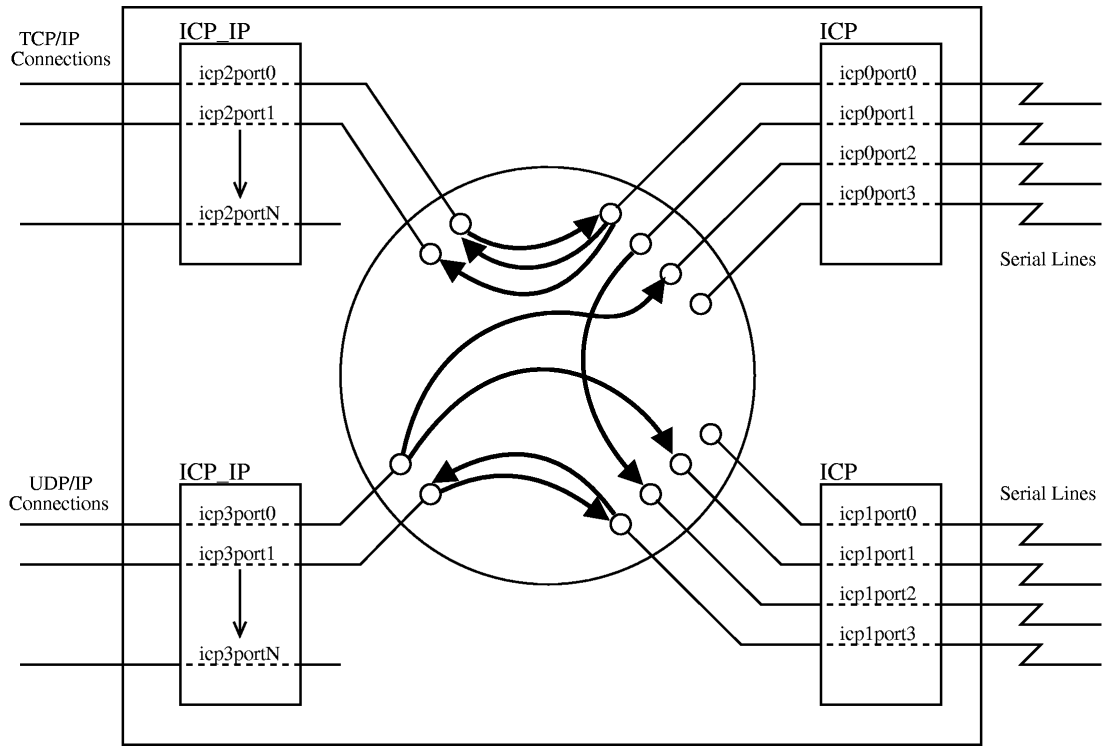


1.3. Message Switch Configuration Example

The Message Switch connects data streams between (and among) ICP devices, somewhat like an old-fashioned plug-in telephone switchboard. Figure 1-2 shows an example Freeway configured with four ICP devices:

1. Two WAN ICP boards (icp0 and icp1)
2. Two ICP_IP pseudo-devices (icp2 and icp3)

Figure 1-2. Example Message Switch Configuration



The Message Switch is shown as a set of arrows in the center of the circle. These arrows represent the switching paths of various data streams (icp0port0 to icp2port0, for example).

The Freeway and the Message Switch use several configuration files to specify the setup of various aspects of the system. These configuration files are described in more detail in Section 2.2. In the example system shown in Figure 1-2 the configuration is as follows:

`bootcfg`

This Freeway boot configuration file specifies the number, type, and characteristics of all ICP devices. Note that the term "ICP devices" includes both physical ICP boards (which communicate via WAN links) and ICP_IP devices (which communicate via TCP/IP or UDP/IP, on an IP connection such as an Ethernet). In Figure 1-2, the `bootcfg` file has specified four ICP devices.

`muxcfg`
`swtcfg`

These two Transport Subsystem Interface (TSI) configuration files specify the transport layer

between the Message Switch and the Freeway. In Figure 1-2 these two files have specified a shared-memory transport (just as the Simpact-supplied files `muxcfg.sra` and `swtcfg` do; most configurations of the Message Switch should be able to use these supplied files without modifications).

`swdcfg`

This DLI configuration file specifies the characteristics of each individual data link (baud rate, clock source, etc.). In Figure 1-2 the swdcfg file has defined 12 data links: icp0port0 through icp3port1.

`switch.cfg`

This main message switch configuration file specifies the inter- connections between/among data links. In Figure 1-2, the `switch.cfg` file has defined 8 connections, each depicted as an arrow from one data link to another.

In summary, the bootcfg file defines the ICP devices, the muxcfg and swtcfg files define the transport layer between the Message Switch and Freeway, the swdcfg file defines the characteristics of each data link, and the switch.cfg file defines the interconnections between data links.

1.4. DLI Concepts for the Message Switch

The Message Switch software uses Simpact's Data Link Interface (DLI), which is a high-level, session-oriented application programming interface (described in the Freeway Data Link Interface Reference Guide).

There are several DLI configuration concepts that you must keep in mind when using the Message Switch software:

- The DLI configuration file needs to include only those session parameters whose values differ from the defaults. Appendix B shows an example `swdcfg` DLI configuration file and `swtcfg` TSI configuration file for the Message Switch.
- The Message Switch software uses DLI Normal operation. Therefore, each session must have the "protocol" DLI configuration parameter set to a specific protocol; for example:

```
protocol = "FMP"
```

- If you are using a Simpact protocol in conjunction with the Message Switch software, DLI Normal operation allows you to define protocol-specific link configuration options in the DLI configuration file. Figure B-4 shows an example session configuration for FMP. Refer to your particular protocol Programmer's Guide for complete link configuration option details.

Chapter 2. Message Switch Configuration and Startup

2.1. Overview

> **Note::** To use the Message Switch software product for Freeway, you must first purchase and install the Freeway software product. Freeway installation is described in the Hardware Installation Guide for your Freeway model and in the Freeway User's Guide.

The Message Switch is a server-resident application (SRA) which runs in a Freeway server. It implements a switching function which transfers all data received by a set of client application sessions to one or more other sessions.

There is no restriction on the type of sessions which can be switched other than that they use DLI Normal operation. This allows data to be switched between ICP devices of different types (for example, ICP2432, ICP2424, or even ICP_IP devices), using different protocols, at different data rates, etc. Data from several sources can also be switched simultaneously to a single destination, or data from a single source may be switched to several destinations, in any combination.

2.2. Configuration Files

The five configuration files listed below must be present in the `freeway/boot` directory to tell the Message Switch how to switch data. The product distribution includes example configuration files containing comments describing the syntax and function of the commands allowed by each configuration file (these files are also shown in Appendix A and Appendix B). Together, these files configure Freeway so that the images built with the `freeway/client/test/makeapi` script can be used to test various features of the Message Switch.

`bootcfg.sw`

The Freeway configuration file that specifies the characteristics of each ICP device (for ICP_IP devices, for example, it specifies the local IP address, foreign IP address, port numbers, etc.). See Section B.1 for an example. Refer to the Freeway User's Guide for a description of the options available in this configuration file.

`swtcfg`
`muxcfg.sra`

The Message Switch TSI configuration file and the Freeway TSI configuration file that specify the shared-memory connection between the Message Switch and Freeway. See Section B.3 for an example. Also refer to the Freeway Transport Subsystem Interface Reference Guide.

`swdcfg`

The Message Switch DLI configuration file that specifies the protocol-level characteristics of each session (baud rate, clock source, etc.). Each session name specifies a particular ICP and link; for example, icp0port0. See Section B.2 for an example. Also refer to the Freeway Data Link Interface Reference Guide and your specific protocol programmer's guide for descriptions of the options available.

`switch.cfg`

The top-level Message Switch configuration file that specifies which source sessions to connect to which destination sessions. The links (specified as session names) must correspond to entries in the DLI configuration file (`swdcfg`). The `switch.cfg` file is detailed in Appendix A.

Also included with the product are three object files, each customized to run with a different type of Freeway:

`sw486.o`

For 486-based Freeway (fw486 image)

`sw68k.o`

For 68k-based Freeway (fw162 image)

`swppc.o`

For ppc-based Freeway (fw2604 image)

In addition to the configuration files and the object files, the distribution includes:

`client/test/switch/apinull.c`

Message Switch test program

`client/test/switch/makeapi`

Message Switch build script

To use the Message Switch, follow the "Quick Start and Test" procedure in the next section. After you reboot your Freeway, the Message Switch starts running. It configures itself automatically, using the setup specified in your configuration files.

2.3. Quick Start and Test of the Message Switch

To quickly get the Message Switch software running and tested, perform the following steps:

> **Note:** You need a Freeway server and another test machine (referred to as Machine "A") which is connected to the same IP network as Freeway. Machine "A" must be capable of compiling and running the apinull.c test program, which uses sockets to read and write packets to the IP network.

> **Note:** At any time, executing the "switchLogOn" command at the Freeway command shell turns on logging of all messages switched by the Message Switch. The "switchLogOff" command turns the logging off.

1. Copy the three Message Switch object files (`sw486.o`, `sw68k.o`, and `swppc.o`) to your Freeway boot directory.
2. Copy the example Message Switch configuration files (`bootcfg.sw`, `switch.cfg`, `swdcfg`, and `swtcfg`) to your Freeway boot directory.
3. Change each occurrence of the string "192.168.135.22" in the file bootcfg.sw to the IP address of machine "A".
4. Add a default route line to the bootcfg.sw file. For example:

```
route_add = 0.0.0.0 192.168.135.22
```

5. Boot the Freeway, being sure that the boot parameters in your Freeway boot directory specify

```
Configuration File Name : bootcfg.sw
```

6. Notice a lot of configuration and status messages produced by the Message Switch immediately after the "FREEWAY SYSTEM ONLINE" message appears. The final message should say:

```
"STATUS:  All Message Switch sessions ready."
```

7. Copy the Message Switch test program and build script to machine "A". These are the two files "`makeapi`" and "`apinull.c`", in the `freeway/client/test/switch/` directory.
8. Change each occurrence of the string "192.168.135.1" in the file `apinull.c` to the IP address of your Freeway.
9. Execute the test program build script on the test machine (machine "A"). For example, if it is a UNIX machine, type "**`./makeapi`**". This builds four programs: "`apinull0`", "`apinull1`", "`apinull2`", and "`apinull3`", which will send/receive packets to the Freeway icp1port0 through icp1port3.
10. On machine "A", run the apinull0 program. This sends a small packet to the icp1port0 device on the Freeway, which the Message Switch receives and returns (via icp1port0) directly back to the apinull0 program. Section A.8.1 shows the `switch.cfg` file specifications.
11. On machine "A", run the apinull1 program. This sends a small packet to the icp1port1 device on Freeway, which the Message Switch receives and sends to several other places (as specified in the switch.cfg file), in addition to returning it (via icp1port1) directly back to the apinull1 program. Section A.8.2 shows the switch.cfg file specifications.
12. On machine "A", run the apinull2 program. This sends a small packet to the icp1port2 device on Freeway, which the Message Switch receives and sends through the icp2/icp3 connection (as specified in the `bootcfg.sw` and the `switch.cfg` files). When it receives the message packet (from icp3), it sends it (via icp1port2) to the apinull2 program. Section A.8.3 shows the switch.cfg file specifications.

To test the TCP/IP listening capabilities of the Message Switch and Freeway (see also Section A.8.5):

13. Open three separate terminal (xterm) windows.
14. In the first window, telnet to port 13824 of Freeway. For example, "telnet freeway0 13824". If you have previously run the apinull1 test program, this telnet window will immediately receive the string "abcdefghijklmnopqrstuvwxyz".
15. In the second window, telnet to port 13825 of Freeway. For example, "telnet freeway0 13825". If you have previously run the apinull1 test program, this telnet window will immediately receive the string "abcdefghijklmnopqrstuvwxyz".
16. In the third window, telnet to port 13826 of Freeway. For example, "telnet freeway0 13826". This telnet window will not receive data immediately, unless you have typed any characters in the first or second telnet sessions.

17. Type "**`telnet 1`**" in the first telnet window, followed by a carriage return. This string should be received by both of the other two telnet sessions.
18. Type "**`telnet 2`**" in the second telnet window, followed by a carriage return. This string should be received only by the telnet session in the first window.

To test the TCP/IP connecting capabilities of the Message Switch and Freeway (see also Section A.8.6):

19. Copy the ICP_IP loopback test program and build script to a separate directory in machine "A". These are the two files "`makeloop`" and "`loopback.c`" in the `freeway/client/test/iploop` directory.
20. Execute the loopback program build script on machine "A". For example, if it is a UNIX machine, type "**`./makeloop`**". This builds 12 programs:

    tcploop2110 tcploop2112 tcploop2114
    tcploop2120 tcploop2122 tcploop2124
    udploop2130 udploop2132 udploop2134
    mcastloop2140 mcastloop2142 mcastloop2144

21. On machine "A", run the tcploop2120 program. This opens two listening TCP sockets to which Freeway can connect. After a minute or two (you must wait until Freeway retries to connect), you should see the tcploop2120 program report "`sockA and sockB ready`".
22. Open two separate terminal (xterm) windows.
23. In the first window, telnet to port 13828 of Freeway. For example, "**`telnet freeway0 13828`**".
24. In the second window, telnet to port 13829 of Freeway. For example, "**`telnet freeway0 13829`**". If you have previously run the "`apinull1`" test program, this telnet window may immediately receive the string "abcdefghijklmnopqrstuvwxyz".
25. Type "**`telnet 3`**" in the first telnet window, followed by a carriage return. This string should be received by the telnet session in the second window.
26. Type "**`telnet 4`**" in the second telnet window, followed by a carriage return. This string should be received by the telnet session in the first window.

If you have an ICP board in your Freeway, and can load it with a protocol, you can also:

27. Uncomment (and modify, if necessary) the lines in `bootcfg.sw` which define icp0 (Figure B-1).
28. Uncomment the lines at the end of `switch.cfg` which specify a switch connection between icp1port3 and icp0port2/3 (Section A.8.4).
29. Install a loopback cable between port2 and port3 of icp0.

30. If necessary, modify the definitions of the icp0port2 and icp0port3 sections of the swdcfg configuration file to match the protocol and type of loopback cable you have (internal/external clocking, etc.).
31. Reboot your Freeway.
32. On machine "A", run the `apinull3` program. This will send a small packet to the icp1port3 device on Freeway, which the Message Switch receives and sends to icp0port2. The loopback cable causes the message to arrive at icp0port3, where the Message Switch receives it and sends it (via icp1port3) back to the `apinull3` program. Section A.8.4 shows the `switch.cfg` file specifications.

If you have an ICP_IP protocol module [such as Simpact's Translator for FMP (TFM) product] you can also:

33. Uncomment (and modify, if necessary) the line toward the end of `bootcfg.sw` which loads your protocol translator module (see Figure B-1). For example,

```
"sra_module = ipfmp486.o"
```

34. Change the lines which specify ipnull in the `bootcfg.sw` file to specify your protocol module for icp2 and icp3, and/or in icp4 and icp5 (see Figure B-1). If you have the TFM product you would change the lines to (for example):

```
"internal_protocol = ipfmp"
```

35. Reboot Freeway.
36. Test with the appropriate protocol loopback program (for example, for the ipfmp protocol module you could test loopbacks on icp2port0 and icp2port1 with the `fmpalp` test program).

To test sending and receiving of multicast packets, you can also:

37. If your Freeway has more than one ethernet interface (you have uncommented and modified the "added_interface_xxx" lines at the end of bootcfg.sw), you may also have to add a "route_add" line to the bootcfg.sw file, to specify which interface the multicast packets should be sent through (as in the commented-out route_add example toward the end of `bootcfg.sw`). See Figure B-1.
38. Uncomment the two lines in `switch.cfg` which specify a link between icp5port0 and icp5port1 (see Section A.8.2).
39. Change the three lines in `switch.cfg` which specify the link between icp1port2 and the icp2/icp3 connection to use the icp4/icp5 connection instead (a commented-out example is shown in that file;

you can simply comment-out the icp2/icp3 lines, and uncomment the icp4/icp5 lines). (see Section A.8.3).

40. Reboot Freeway.
41. On machine "A", run the apinull2 test program to test the icp4/icp5 connection (which uses multicast). Section A.8.3 shows the `switch.cfg` file specifications.
42. If you are using a protocol module other than ipnull, you can also test with the appropriate protocol loopback program on icp4port0 and icp4port1, as in test step 36 above. It may be necessary to modify the loopback test program to allow use of icp4, and to add definitions for icp4port0 and icp4port1 to the loopback test program's DLI configuration file.

2.4. ICP_IP Devices

Note: The information in this section is included in the Freeway User's Guide and is reproduced here because the Message Switch is particularly useful with ICP_IP devices.

The ICP_IP devices which have been added to Freeway's capabilities behave similarly to any ordinary ICP device (ICP6000, ICP2432, etc.). The desired configuration of an ICP_IP device must be specified in the boot configuration file (this is the file which is specified as the "Configuration File Name" field of the Freeway boot parameters). An example configuration file containing ICP_IP configuration specifications is provided with the Freeway software release (called `bootcfg.ip`). See that file for a description of the configuration options which have been added to support ICP_IP.

The steps required to use the new ICP_IP devices are:

1. Edit `bootcfg.ip` to change the IP addresses as appropriate for your system.
   - All references to "192.168.135.1" should be changed to the IP address of your Freeway system
   - All references to "192.168.135.22" should be changed to the IP address of a system (or systems) which will send or receive IP packets to or from your Freeway system
   - If you plan to use the second ethernet interface, all references to "192.168.136.1" and "192.168.136.22" must also be changed

2. Change the following boot parameters for your Freeway system:
   - Change the "FREEWAY Server Name" and "FREEWAY Inet Address" to tell the Freeway system where it is configured in your network.

- Change the "Boot Server Name" and "Boot Server Inet Address" to specify a machine from which the Freeway boot directory can be accessed.
- Change the "System Boot Directory" to point to your Freeway boot directory.
- Change the "System Boot File Name" to the name of your Freeway image.
- Change the "Configuration File Name" to bootcfg.ip.

3. Reboot Freeway.
4. The new ICP_IP devices can now be used in either raw or normal mode by any client application, simply by specifying the appropriate ICP name in the call to dlOpen. For example, if Freeway is configured as shown in the supplied `bootcfg.ip` file, a call to dlOpen for a port definition with BoardNo set to 3 will use a socket configured as specified in icp3 in `bootcfg.ip`.

   The mapping looks like this:

   - The dlOpen function is called with a session name. For example: server0icp3port4
   - The session name is matched to a section in the DLI configuration file, and the appropriate BoardNo and PortNo values are saved. For example:

     ```
     BoardNo = 3
     PortNo = 4
     ```

   - The BoardNo, appended to the ascii string "icp", is used to find the device specified in the Freeway boot configuration file. The device specifies the type of data transfer to be used, local and foreign IP addresses, base port numbers, etc. The PortNo from the DLI configuration file is added to the local_port_base and foreign_port_base fields from the boot configuration file to generate the port numbers used for this IP device.

     For example:

     ```
     device = icp3
     type = sock_dgram
     local address = ANY
     local port = 0x2130 + 4 = 0x2134
     foreign address = 192.168.135.22
     foreign port = 0x2130 + 4 = 0x2134
     ```

- To test an ICP_IP link, you may run the "`makeloop`" script on a UNIX machine to compile the loopback test programs. This script and the accompanying `loopback.c` source file reside in the directory `freeway/client/test/iploop`. These programs perform the same function for ICP_IP links as the physical loopback cables perform for real wired FMP links; they take data being sent to one ICP_IP link and return it to the next adjacent ICP_IP link.
  - Make sure the Freeway system has an ICP_IP device pointing to the UNIX machine where the test program is running, (change the foreign_address to the IP address of the machine where the test program is running) then run any normal test program (for example, `fmpalp`).
  - The loopback test programs correspond to the ICP_IP devices as configured in the supplied `bootcfg.ip` file:

```
server0icp1port0 (and port1) tcploop2110
server0icp1port2 (and port3) tcploop2112
server0icp1port4 (and port5) tcploop2114

server0icp2port0 (and port1) tcploop2120
server0icp2port2 (and port3) tcploop2122
server0icp2port4 (and port5) tcploop2124

server0icp3port0 (and port1) udploop2130
server0icp3port2 (and port3) udploop2132
server0icp3port4 (and port5) udploop2134

server0icp4port0 (and port1) mcastloop2140
server0icp4port2 (and port3) mcastloop2142
server0icp4port4 (and port5) mcastloop2144
```

Note: The mcastxxxx programs must be compiled and run on a multicast-capable machine, and are necessary only for testing a multicast-capable Freeway.

For example, if you set foreign_address for icp3 to the IP address of a machine running udploop2134, you can run fmpalp on any client machine (with the "server" field in fmpaltcfg pointing to your Freeway), select "3" when asked for the "ICP board on which to run test", select "4" when asked for the "Even port number", and fmpalp will run normally. The data will be going from fmpalp on your client machine, to Freeway, out the ICP_IP device to the machine running udploop2134, back to Freeway on the alternate-numbered ICP_IP link, then back to fmpalp on your client machine.

Appendix A. Message Switch switch.cfg File

The top-level Message Switch configuration file is named `switch.cfg`. The entire example file is duplicated in this Appendix but is broken into sections to illustrate specific types of configuration scenarios. In all the examples, the link names (icp0port0, etc.) correspond to the session names specified in the example `swdcfg` file (shown in Figure B-2) supplied with the Message Switch product.

Section A.1 through Section A.5 illustrate specifications that the Message Switch performs autonomously, without any client session connection involved.

Section A.6 through Section A.8 illustrate the simultaneous use of client sessions, which are of three types:

- Section A.6 and Section A.7 illustrate traditional Freeway DLI/TSI client sessions, which are allowed as long as they do not attempt to connect to a device and port which is already in use by the Message Switch. (For example, if the Message Switch is connected to icp1port0, then no other client can use that device.)
- Section A.8.1 through Section A.8.4 illustrate simple socket client connections using UDP/IP sockets.
- Section A.8.5 and Section A.8.6 illustrate simple socket client connections using TCP/IP sockets.

A.1. General Description

```
#-------------------------------------------------#
#                                                 #
#  This is the main configuration file for the Freeway Message Switch.     #
#  It contains commands which cause the Message Switch task to             #
#  copy all data received on one link to another link or links.            #
#                                                 #
#  Blank lines, and lines which have a "#" as the first character, are     #
#  treated as comments, and have no effect on the run-time operation       #
#  of the Message Switch task.                                             #
#                                                 #
#  The first non-comment line in this file must be of the form:            #
#                                                 #
#     DLICfgFile = <filename>                                              #
#                                                 #
#  Where the <filename> specifies the name of a file which will be         #
#  used as the DLI configuration file for the Message Switch task.         #
#  If no path name is provided in the <filename> field, and the            #
#  switch task is executing as an SRA on a Freeway, then the switch        #
#  task will search the System Boot Directory specified in the             #
#  Freeway System Boot Parameters for the configuration file.  A           #
#  fully qualified path name to the configuration file may be used         #
#  if desired.                                                             #
```

```
#                                                                      #
#  All lines following the specification of the DLI configuration file #
#  must be either blank lines, comments, or switch commands of the     #
#  following form:                                                     #
#                                                                      #
#     read_link > write_link                                           #
#                                                                      #
#  or                                                                  #
#                                                                      #
#     read_link » write_link                                           #
#                                                                      #
#  Switch commands will cause the Message Switch task to open the two  #
#  specified links, then continually execute read commands on the link #
#  specified to the left of the ">" or "»" string, and write the       #
#  resulting data to the link specified to the right of the ">" or "»" #
#  string.  The links specified must correspond to entries in the DLI  #
#  configuration file for the Message Switch task.                     #
#                                                                      #
#  It is permissible to specify the same read link on multiple command #
#  lines; the result will be that data read from that link will be     #
#  written to all of the specified write links.                        #
#  It is also permissible to specify the same write link on multiple   #
#  command lines; the result will be that data read from multiple      #
#  read links will be written to the specified write link.             #
#                                                                      #
#  If it is desired that all data in both directions be switched       #
#  between two links, two switch commands will be necessary, as        #
#  shown in this example:                                              #
#                                                                      #
#     icp0port0 > icp1port0                                            #
#     icp1port0 > icp0port0                                            #
#                                                                      #
#  If a switch command uses a single ">" character between link        #
#  specifiers, then the Message Switch will perform flow control; it   #
#  will stop reading from the source session if the destination session #
#  becomes unable to receive further data.                             #
#                                                                      #
#  If a switch command uses the double-">" string ("»") between link   #
#  specifiers, then the Message Switch will not perform flow control;  #
#  it will continue to read from the source and attempt to write to the #
#  destination regardless of the destination's ability to accept data. #
#                                                                      #
#  Note that any flow-controlled destination which blocks will cause the #
#  Message Switch to stop reading from a source session, so if there are #
#  several connection definitions for a single source (specifying      #
#  that data from that source should go to multiple destinations) and  #
#  some of those connections specify flow-control, then all destinations #
#  will stop receiving data when any flow-controlled destination blocks. #
#                                                                      #
#  If the destination session is a UDP ICP_IP device, then the flow    #
#  control setting has no practical effect, since UDP packet streams   #
#  never block.                                                        #
#                                                                      #
#----------------------------------------------#
```

```
#                                                                       #
#  If the Message Switch task is to run as an SRA in a Freeway, a       #
#  command of the following form must be placed in the boot             #
#  configuration file (bootcfg.xxx):                                    #
#                                                                       #
#    sra_module = <name of object file containing switch task>          #
#    sra_entry  = <name of switch task entry point to be spawned>       #
#                                                                       #
#  For example:                                                         #
#                                                                       #
#    sra_module = sw486.o                                               #
#    sra_module = sraSwitch.o                                           #
#                                                                       #
#  The "tsi_config_file = muxcfg" line in the boot configuration file   #
#  must also be changed to "tsi_config_file = sramuxcfg", (or to point  #
#  to another muxcfg file which defines a shared-memory connection).    #
#                                                                       #
#-----------------------------------------------#
DLICfgFile = swdcfg
# In the examples below, the link names (icp0port0, etc.) correspond to the
# names specified in the example swdcfg file supplied with the Message Switch.
```

A.2. Direct Reply "Ping" Switch Specification

```
# Here is an example of a simple direct reply "ping" switch specification.
# All data read from icp1port0 will be written back to icp1port0.
#
# icp1port0 > icp1port0
```

A.3. Two-way Switch Between Two Data Streams Specification

```
# Here is an example of a simple 2-way switch between 2 data streams.
# All data in both directions will be switched between icp0port1 and
# icp1port1.
# Note that icp0 and icp1 may be running different protocols; the data
# switching will still work as long as each is a Simpact-supported
# "normal"-mode protocol, and the swdcfg configuration file specifies a
# compatible data transfer mode between the two (for example, if one is
# the FMP protocol, it should have 'MessageBlocking = "DataBlk";').
#
# icp0port1 > icp1port1
# icp1port1 > icp0port1
```

A.4. One-to-many Specification

```
# Here is an example of a single source going to multiple destinations.
# All data from icp1port2 will be sent to all the specified destinations.
# Again, icp1, icp6, and icp7 may all be running different protocols.
#
# icp1port2 > icp6port0
# icp1port2 > icp6port1
# icp1port2 > icp7port0
```

A.5. Many-to-One Specification

```
# Here is an example of multiple sources going to a single destination.
# It is also possible to cause a message from a single source to be sent
# twice or more to the same destination (if the line "icp0port0 > icp6port3"
# were repeated several times, for instance).
#
# icp0port0 > icp6port3
# icp1port0 > icp6port3
# icp2port0 > icp6port3

#-------------------------------------------------#
# The examples above illustrate switch specifications which the           #
# Message Switch will perform autonomously, without any client            #
# session connection; it simply switches packets between data streams.    #
# Client sessions may be used simultaneously, as long as no client        #
# sessions attempts to connect to a device and port which is already in   #
# use by the Message Switch (for example, if the Message Switch is        #
# connected to icp1port0, then no other client may use that device).      #
#                                                                         #
```

A.6. Client Applications Connecting to the Message Switch

```
# It is also possible to setup the ICP configuration in the bootcfg       #
# file to allow client applications to send and receive data packets      #
# to and from the Message Switch.  This allows client applications to     #
```

```
# take advantage of all the functionality provided by the Message Switch   #
# (one-to-many, many-to-one, etc.)  To configure this in the bootcfg file, #
# an IP protocol module must be used with two "ICP_IP" devices.             #
# The ICP_IP devices should be set up with lines similar to this, in        #
# the boot configuration file (this assumes that the client application     #
# will use ’Protocol = "FMP"’ in its DLI configuration file):               #
#                                                                           #
#    device_name          =  icp2                                          #
#    device_type          =  icp_ip                                        #
#    socket_type          =  sock_dgram                                    #
#    local_address        =  0.0.0.0                                       #
#    local_port_base      =  0x3200                                        #
#    foreign_address      =  127.0.0.1                                     #
#    foreign_port_base    =  0x3300                                        #
#    internal_protocol    =  ipfmp                                         #
#                                                                          #
#    device_name          =  icp3                                          #
#    device_type          =  icp_ip                                        #
#    socket_type          =  sock_dgram                                    #
#    local_address        =  0.0.0.0                                       #
#    local_port_base      =  0x3300                                        #
#    foreign_address      =  127.0.0.1                                     #
#    foreign_port_base    =  0x3200                                        #
#    internal_protocol    =  ipfmp                                         #
#                                                                          #
#    sra_module           =  ipfmp486.o                                    #
#                                                                          #
# Notice that icp2port0 writes to port 0x3300 and reads from 0x3200,       #
# while       icp3port0 writes to port 0x3200 and reads from 0x3300.       #
# All data written to icp2port0 will be received at icp3port0, and         #
# vice-versa.                                                              #
#                                                                          #
# Thus, if the Message Switch is configured to connect to icp3port0,       #
# it will receive all data written by any client application to            #
# icp2port0, and the client application will receive everything that       #
# the Message Switch writes to icp3port0.                                  #
#                                                                          #
#--------------------------------------------------#
#
# With the above lines in the bootcfg file, Message Switch
# specifications like:

#
# icp3port0 > icp6port0
# icp3port0 > icp6port1
# icp3port0 > icp7port0

#
# could be used to cause all data written by a client program
# to icp2port0 to be written to all the links shown.  Or,
# Message Switch specifications like:

#
# icp1port0 > icp3port0
```

```
# icp1port1 > icp3port0
# icp2port0 > icp3port0

#
# would cause all data received by the Freeway on the 3 links shown
# to be sent to icp3port0, where they will be received by a single client
# program connected to icp2port0.  Or, Message Switch specifications like:

#
# icp1port0 > icp3port0
# icp1port0 > icp3port1
# icp1port0 > icp3port2

#
# would cause each data packet received by the Freeway on icp1port0 to be
# sent (through the icp3-icp2 connection) to 3 different client applications,
# each connected to a separate port on icp2.
#
```

A.7. Using Multicast-Capable Freeway

```
#-------------------------------------------------#
# If you are running Simpact's Multicast-capable Freeway server           #
# software, there is another way to connect two or more data streams      #
# to a client application (or applications).  This takes advantage of     #
# the fact that the Freeway can receive Multicast data packets even       #
# if it was the originator of those packets.  The bootcfg file might      #
# look like this (this also shows the IPFMP protocol module):             #
#                                                                         #
#    device_name          =  icp4                                         #
#    device_type          =  icp_ip                                       #
#    socket_type          =  sock_dgram                                   #
#    local_address        =  0.0.0.0                                      #
#    local_port_base      =  0x3400                                       #
#    foreign_address      =  234.1.1.2                                    #
#    foreign_port_base    =  0x3500                                       #
#    internal_protocol    =  ipfmp                                        #
#                                                                         #
#    device_name          =  icp5                                         #
#    device_type          =  icp_ip                                       #
#    socket_type          =  sock_dgram                                   #
#    local_address        =  234.1.1.2                                    #
#    local_port_base      =  0x3500                                       #
#    foreign_address      =  234.1.1.3                                    #
#    foreign_port_base    =  0x3400                                       #
#    internal_protocol    =  ipfmp                                        #
#                                                                         #
#    sra_module           =  ipfmp486.o                                   #
#                                                                         #
```

```
# Then Message Switch specifiers like:                                  #
#                                                                       #
#    icp5port0 > icp5port0                                              #
#    icp5port0 > icp6port0                                              #
#                                                                       #
# would mean that a client application writing a single packet to       #
# icp4port0 would cause an IPFMP packet to be sent to 3 places:         #
#                                                                       #
#    1) 234.1.1.2 port 0x3500 (written directly by the client app)      #
#    2) 234.1.1.3 port 0x3400 (written by Message Switch to icp5port0)  #
#    3) whereever icp6port0 is configured for                           #
#                                                                       #
#-----------------------------------------------#
```

A.8. Using the Null ICP_IP Device

```
#-----------------------------------------------#
# Another way to use the Message Switch is to use the ipnull ICP_IP     #
# device directly.  This allows an extremely simple client application  #
# program, written without any calls to DLI, to send and receive data  #
# from a Freeway.  Here's how the bootcfg file might be set up:         #
#                                                                       #
#    device_name          =  icp1                                       #
#    device_type          =  icp_ip                                     #
#    socket_type          =  sock_dgram                                 #
#    local_address        =  0.0.0.0                                    #
#    local_port_base      =  0x3100                                     #
#    foreign_address      =  192.168.100.100                            #
#    foreign_port_base    =  0x3100                                     #
#    internal_protocol    =  ipnull                                     #
#                                                                       #
# And the switch.cfg file contained:                                    #
#                                                                       #
#    icp1port0 > icp0port0                                              #
#    icp0port0 > icp1port0                                              #
#                                                                       #
# Then an application program running on a machine at 192.168.100.100   #
# could simply create a socket, bind to port 0x3100, and send UDP       #
# datagrams to the IP address of the Freeway; the Message Switch        #
# would send that data to icp0port0 to be sent out with whatever        #
# protocol settings were specified for that port in the swdcfg file     #
# (and any data received on that port would be sent to the UDP client   #
# application program at 192.168.100.100).                              #
#                                                                       #
# The UDP datagrams must follow the format specified for the protocol   #
# module loaded into the corresponding ICP_IP device.  For the ipnull   #
# protocol, which is builtin to all Freeway images, there is no format; #
# it simply treats each UDP packet as a block of raw data, to be packaged #
# up and sent to the Message Switch.  This makes it extremely easy to   #
```

```
# write application programs which send and receive data to/from a         #
# Freeway.  A very short example program which does just that is in        #
# freeway/client/test/switch/apisim.c .  This is the program which can be  #
# used to create the apinull0, apinull1, apinull2, and apinull3            #
# executables which are used to test the Message Switch, as shown in the   #
# examples below.                                                          #
#                                                                          #
#--------------------------------------------------#
#
#
#
#--------------------------------------------------#
#                                                                          #
# EXAMPLES:                                                                #
#                                                                          #
# The following example switch specifications can be used with the         #
# bootcfg.sw file provided with the Message Switch to test various         #
# configurations.                                                          #
# They expect that all data sent to the Freeway will be from either a      #
# DLI client using FMP, or from the "apinull" sample programs which are    #
# also included with this software.                                        #
#                                                                          #
#--------------------------------------------------#
```

A.8.1. UDP/IP Example (apinull0)

Note: The example program apinull0 is supplied for testing the Message Switch (see test step 10 in section Section 2.3).

```
#
# apinull0 can be used to send packets to icp1port0; this switch
# specification causes each packet received to be transmitted back to
# apinull0.

icp1port0 > icp1port0
```

A.8.2. UDP/IP Example (apinull1)

Note: The example program apinull1 is supplied for testing the Message Switch (see test step 11 in section Section 2.3).

```
# apinull1 can be used to send packets to icp1port1; these switch
# specifications cause each packet received to be transmitted to 3 other
# places, in addition to being transmitted back to apinull1.

icp1port1 > icp6port0
icp1port1 > icp6port1
icp1port1 > icp7port0
icp1port1 > icp1port1

# these switch specifications allow a normal loopback test program, such
# as fmpalp, to work correctly between icp2port0 and icp2port1, if the
# bootcfg file sets up icp2 and icp3 as shown above, and loads the
# appropriate protocol module into the ICP_IP devices at icp2 and icp3.
# (These 2 specifications take the place of the loopback cable).

icp3port0 > icp3port1
icp3port1 > icp3port0

# these switch specifications allow a normal loopback test program, such
# as fmpalp, to work correctly between icp4port0 and icp4port1, if the
# bootcfg file sets up icp4 and icp5 as shown above, and loads the
# appropriate protocol module into the ICP_IP devices at icp4 and icp5.
# (These 2 specifications take the place of the loopback cable).
# Uncomment these 2 lines if you are using a multicast-capable Freeway
# server, and wish to test the icp4/icp5 link.

# icp5port0 > icp5port1
# icp5port1 > icp5port0
```

A.8.3. UDP/IP Example (apinull2)

Note: The example program apinull2 is supplied for testing the Message Switch (see test step 12 in section Section 2.3).

```
# apinull2 can be used to send packets to icp1port2; these switch
# specifications cause each packet received to be transmitted through the
# icp2/icp3 connection, looped back (the icp3port2 and port3 specifications),
# and returned to apinull2.  The bootcfg file must have set up icp2 and
# icp3 as shown above.

icp1port2 > icp2port2
icp3port2 > icp3port3
icp2port3 > icp1port2

# The same thing could be done using the icp4/icp5 connection:
# icp1port2 > icp4port2
# icp5port2 > icp5port3
```

```
# icp4port3 > icp1port2
```

A.8.4. UDP/IP Example (apinull3)

Note: The example program apinull3 is supplied for testing the Message Switch (see test step 32 in section Section 2.3).

```
# apinull3 can be used to send packets to icp1port3; these switch
# specifications cause each packet received to be transmitted through
# icp0 (which is assumed to have a loopback cable installed between
# icp0port2 and icp0port3), and returned to apinull3.
# To test this configuration, uncomment the 2 switch configuration lines
# here and change bootcfg.sw to include a definition for icp0 which includes
# the loading of the fmp protocol; also make sure there is a loopback
# cable installed between links 2 and 3 of icp0.

# icp1port3 > icp0port2
# icp0port3 > icp1port3
```

A.8.5. TCP/IP "Listening" Connections

Note: The following examples are for testing the TCP/IP listening capabilities of the Message Switch and Freeway (see test steps 13 through 18 in section Section 2.3).

```
# The following switch settings allow testing of TCP listening ICP_IP devices.
# These devices listen at a specified TCP/IP port on the Freeway, waiting
# for client application programs to connect to that port.  For these
# examples, we will use the "telnet" client program to connect to the
# Freeway at the specified ports.

# The following switch settings will cause the Message Switch to switch
# all data between icp6port0 and icp6port1; these are configured in the
# bootcfg.sw file as TCP sockets listening on ports 0x3600 and 0x3601
# (13824 and 13825).  To test, use telnet to connect to those ports on
# the Freeway - for example, if your Freeway IP address is 192.168.135.22,
# then execute these commands from two different windows:
#
#     telnet 192.168.135.22 13824      (in first  window)
```

```
#     telnet 192.168.135.22 13825      (in second window)
#
# Then everything you type in the first window will be copied to the second
# window, and vice-versa.

icp6port0 » icp6port1
icp6port1 » icp6port0

# You can also configure additional ports to receive data.  The following
# switch setting will cause all data you type in your first telnet window
# to be copied to a third telnet window in addition to the second window.
# You can retrieve the data from icp6port2 with the command:
#
#     telnet 192.168.135.22 13826      (in third window)
icp6port0 » icp6port2

# The following switch setting causes all data received by icp3port0 to be
# sent to icp6port3.  To see this data, telnet to your Freeway at the
# icp6port3 port (Simpact settings are 0x3600 + 3 = 0x3603, # or 13827).
# For example:  telnet 192.168.135.22 13827
#
# Then use your normal loopback test program (for example, fmpalp) to
# send data between icp2port0 and icp2port1.  The data sent from icp2port0
# to icp3port0 will be sent by the Message Switch to icp3port1 (where it
# will be returned to icp2port1, to satisfy the loopback program) and to
# icp6port3, where it will be sent to your telnet session.

icp3port0 » icp6port3
```

A.8.6. TCP/IP "Connecting" Connections

Note: The following examples are for testing the TCP/IP connecting capabilities of the Message Switch and Freeway (see test steps 19 through 26 in section Section 2.3). These tests require use of the loopback test program and build script (`loopback.c` and `makeloop`) which are supplied with the Freeway software distribution in the `freeway/client/test/iploop` subdirectory.

```
# The following switch settings allow testing of a TCP connect ICP_IP device.
# These devices connect to a specified TCP/IP port at a specified IP
# address.  For this example we will use the
# freeway/client/test/iploop/tcploop2120 loopback test program, to serve as
# a TCP listener for the Message Switch to connect to.  When tcploop2120 is
# running, and the Freeway has connected to both ports (tcploop2120 will
# report "sockA and sockB ready."), then telnet clients can be used to
# connect to icp6port4 and icp6port5 to send and receive data:
#
#     telnet 192.168.135.22 13828
```

```
#     telnet 192.168.135.22 13829
#
# The data goes from the telnet client (via Message Switch) to icp7port0,
# which sends it to tcploop2120, which returns it to icp7port1, and the
# Message Switch sends it to icp6port4 where it is received by the telnet
# client at icp6port5.  The bootcfg.sw file must be modified to set the
# foreign_address of icp7 to the IP address of the machine where
# tcploop2120 is run.

icp6port4 > icp7port0
icp7port1 > icp6port5

# and the reverse direction of data flow is defined like this:

icp6port5 > icp7port1
icp7port0 > icp6port4
```

Appendix B. Other Message Switch Configuration Files

This appendix shows the three configuration files that support the top-level Message Switch configuration file named `switch.cfg`, which was described in Appendix A. The three support files are:

- bootcfg.sw - Freeway boot configuration file (Section B.1 below)
- swdcfg - DLI configuration file (Section B.2)
- swtcfg - TSI configuration file (Section B.3)

B.1. Freeway Boot Configuration File (bootcfg.sw)

Figure B-1 is an example Freeway Boot Configuration file (bootcfg.sw) for the Message Switch. The bootcfg.sw file is used in the procedures described in Chapter 2.

Figure B-1. Freeway bootcfg.sw File

```
#-----------------------------------------------#
#  bootcfg.sw                                   #
#                                               #
#  This file is intended for use with the Message Switch and its        #
#  delivered configuration files: switch.cfg, swdcfg, and swtcfg.       #
#  Together, these configuration files will configure a Freeway in      #
#  such a way that the executables built with the makeapi script in     #
#  freeway/client/test/switch can be used to test various features of   #
#  the Message Switch.                                                  #
#                                                                       #
#  To configure a Freeway system to use the settings specified here,    #
#  the name of this file must be listed in the Freeway's boot           #
#  parameters under the name "System Boot File Name".                   #
#                                                                       #
#-----------------------------------------------#
#                                                                       #
#  This file is preconfigured with the following devices:               #
#                                                                       #
#    icp0    definition of ICP2432 loaded with FMP (commented-out).     #
#    icp1    ICP_IP (UDP unicast)   loaded with ipnull.                 #
#    icp2    ICP_IP (UDP unicast)   loaded with ipnull.                 #
#    icp3    ICP_IP (UDP unicast)   loaded with ipnull.                 #
#    icp4    ICP_IP (UDP multicast) loaded with ipnull.                 #
#    icp5    ICP_IP (UDP multicast) loaded with ipnull.                 #
#    icp6    ICP_IP (TCP listener)  loaded with ipnull.                 #
#    icp7    ICP_IP (TCP connector) loaded with ipnull.                 #
```

```
#                                                                    #
#  See the other example bootcfg.xxx files delivered with your       #
#  Freeway for descriptions of the syntax and implementation of      #
#  these ICP device configurations.                                  #
#                                                                    #
#  In the example specifications included in this file, the IP       #
#  addresses of the Freeway have been assumed to be 192.168.135.1 for #
#  the primary ethernet and 192.168.136.1 for the secondary, and     #
#  the IP address of the foreign system has been assumed to be       #
#  192.168.135.22 .  These values must be changed to the actual values #
#  for your particular Freeway system and environment.               #
#                                                                    #
#--------------------------------------------#

#-------------------------------------------#
# ICP-0 Physical Parameter example definitions.                  #
# If you have an ICP board in your Freeway, uncomment these lines. #
#-------------------------------------------#

# device_name          =  icp0
# device_type          =  icp2432
# slave_address        =  0x14
# download_script      =  fmpload.2432

#-------------------------------#
# The following parameters define ICP_IP devices #
#-------------------------------#

#-------------------------#
# UDP/IP  ICP_IP data streams.        #
#-------------------------#

#------------------#
# The "null api" interface. #
#------------------#

device_name          =  icp1
device_type          =  icp_ip
socket_type          =  sock_dgram
local_address        =  0.0.0.0
local_port_base      =  0x3100
foreign_address      =  192.168.135.22
foreign_port_base    =  0x3100
internal_protocol    =  ipnull


#---------------------------------------#
# NOTE: icp2 through icp7 are configured in the swdcfg file  #
# for use with either the "NULL" or the "FMP" protocol       #
# module.  If you have the FMP protocol module (Simpact      #
# product FWTFM) installed in your freeway/boot directory    #
# (files ipfmp486.o, ipfmp68k.o, or ipfmpppc.o),             #
# change the internal_protocol from ipnull to ipfmp for any  #
# of the devices ICP2 through ICP7.  You must also           #
```

```
# uncomment the line near the bottom of this bootcfg.sw      #
# file which loads the appropriate ipfmp module              #
# ("sra_module = ipfmp486.o").                               #
#--------------------------------------#

#-------------------------------------#
# Unicast connection between icp2 and icp3.              #
# To enable use of a standard loopback program (such as  #
# fmpalp), change the internal protocol as appropriate   #
# (for example, ipfmp).                                  #
#-------------------------------------#

device_name          =  icp2
device_type          =  icp_ip
socket_type          =  sock_dgram
local_address        =  0.0.0.0
local_port_base      =  0x3200
foreign_address      =  127.0.0.1
foreign_port_base    =  0x3300
internal_protocol    =  ipnull
# internal_protocol    =  ipfmp

device_name          =  icp3
device_type          =  icp_ip
socket_type          =  sock_dgram
local_address        =  0.0.0.0
local_port_base      =  0x3300
foreign_address      =  127.0.0.1
foreign_port_base    =  0x3200
internal_protocol    =  ipnull
# internal_protocol    =  ipfmp


#-------------------------------------#
# Multicast connection between icp4 and icp5.            #
# To enable use of a standard loopback program (such as  #
# fmpalp), change the internal protocol as appropriate   #
# (for example, ipfmp).                                  #
#                                                        #
# NOTE: icp4 and icp5 will only work if the Freeway was  #
# booted with a multicast-capable server image.          #
#-------------------------------------#

device_name          =  icp4
device_type          =  icp_ip
socket_type          =  sock_dgram
local_address        =  234.5.6.4
local_port_base      =  0x3400
# local_if_address     =  192.168.135.1   # local interface to receive mcasts
foreign_address      =  234.5.6.4
foreign_port_base    =  0x3500
ttl                  =  0x1
internal_protocol    =  ipnull
# internal_protocol    =  ipfmp
```

```
device_name          =  icp5
device_type          =  icp_ip
socket_type          =  sock_dgram
local_address        =  234.5.6.4
local_port_base      =  0x3500
# local_if_address     =  192.168.135.1   # local interface to receive mcasts
foreign_address      =  234.5.6.4
foreign_port_base    =  0x3400
internal_protocol    =  ipnull
# internal_protocol    =  ipfmp

#-------------------------#
# TCP/IP  ICP_IP data streams.        #
#-------------------------#

device_name          =  icp6
device_type          =  icp_ip
socket_type          =  sock_stream_listen
local_address        =  0.0.0.0
local_port_base      =  0x3600              # 13824
foreign_port_base    =  0x0000
send_q_size          =  0x14                # 20 packets
internal_protocol    =  ipnull
device_name          =  icp7
device_type          =  icp_ip
socket_type          =  sock_stream_connect
connect_period       =  0x40                # 64 seconds
local_address        =  0.0.0.0
local_port_base      =  0x0000
foreign_address      =  192.168.135.22
foreign_port_base    =  0x2120              # 8480
internal_protocol    =  ipnull


#------------------#
# Server wide parameters    #
#------------------#

tsi_config_file      =  muxcfg.sra

# If uncommented, the following route_add statement will cause all multicast
# packets to be sent to the secondary interface (the interface with IP
# address 192.168.136.1).
# route_add            =  224.0.0.0     192.168.136.1

# The following line shows an example definition for a default route.
# route_add            =  0.0.0.0       192.168.135.10

#               Uncomment the "sra_module = ipfmp486.o" if you have the
#               ipfmp486.o file and want to test the ipfmp protocol.

# sra_module           = ipfmp486.o
#               The following line loads the Message Switch.
```

```
sra_module           = sw486.o
sra_entry            = sraSwitch

vxworks_shell         = TRUE

#               Uncomment these "added_interface_xxx" lines to configure a
#               second ethernet interface.  You may need to add a "route_add"
#               statement similar to the example above to cause the Freeway
#               to send the packets through the desired interface.

# added_interface_type =  fei
# added_interface_mask =  ffffff00
# added_interface_addr =  192.168.136.1
```

B.2. DLI Configuration File (swdcfg)

Figure B-2 through Figure B-7 illustrate the `swdcfg` file used in the procedures described in Chapter 2. Each of the following figures illustrates a particular section of the `swdcfg` file. This is a partial example (since many session definitions are similar). Refer to the actual product swdcfg file for complete details. The Figure B-2 "Main" section parameters apply to all sessions defined in the remainder of the file.

Figure B-2. DLI Configuration File: "Main" Section

```
//-------------------------------------------------
//
// swdcfg
//
// This is an example DLI configuration file for use by the Message Switch,
// an SRA which switches data packets between DLI sessions within a Freeway.
// The switch.cfg file (the main configuration file for the Message Switch),
// must have a line which points to this file - for example:
//
//    DLICfgFile = swdcfg
//
// The switch.cfg file specifies which sessions should be switched to which
// other sessions; this file defines the characteristics of each of those
// sessions.
//
// This file uses many of the default DLI settings, which are not explicitly
// specified here; see the DLI Reference Guide for a description of all DLI
// configuration parameters.
//
//-------------------------------------------------

//-------------------------------------------------
// "main" section.  When present, the main section must appear first in the
// DLI configuration file.
```

```
//
// Note:
// For the Message Switch, the tsi config file specified in the main section
// below _MUST_ be "sw0:swtcfg.bin", and the name of the file in the Freeway
// boot directory must be "swtcfg" .
//
//-------------------------------------------------

main
{  TSICfgName = "sw0:swtcfg.bin";       // TSI binary configuration file name.
  AsyncIO    = "Yes";                  // SRAs must use asynchronous I/O.
//  LogLev          = 7;
  SessPerConn = 4;
}
```

Figure B-3 defines the protocol-independent parameters for the first session, which is named "icp0port0."

Figure B-3. Session "icp0port0": Protocol-independent Parameters

```
// The following definitions are used by the Message Switch program.

//-------------------------------------------------
// ICP_2432 device.
// This is for use by the Message Switch, to route data to an ICP board.
// This example assumes ICP0 is loaded with the FMP protocol.
//-------------------------------------------------
icp0port0
{                              // messageblocking must be "datablk"
                               // for the Message Switch, so that the data
                               // read from the device is in the exact same
                               // format as data to be written.
  // ---------------------
  // Protocol-independent parameters.
  // ---------------------
  Protocol           = "FMP";          // FMP session
  Transport          = "Conn0";        // Transport connection name
                                       //    defined in file swtcfg.
  // Family          = "Protocol";
  BoardNo            = 0;              // ICP board number zero.
  PortNo             = 0;              // Port number zero.
  // PortNo             = 0x'2014';         // IP Port number.
  Timeout            = 100;            // 100 seconds.
  // LogLev          = 0;
  // TraceLev        = 0;
  // MaxInQ          = 10;
  // MaxOutQ         = 10;
  // MaxErrors       = 100;
  // LocalAck        = "Yes";
  // CfgLink         = "Yes";
  // ReuseTrans      = "No";
  AsyncIO            = "Yes";          // Asynchronous I/O.
```

```
  // OutOfBand       = "No";
  // OldBind         = "Yes";
  AlwaysQIO          = "Yes";          // Comply with POSIX standard.
  // Segmenting      = "No";
  // Buffering       = "No";
  // Server          = "No";
```

Figure B-4 defines the FMP protocol-specific parameters for the first session, which is named "icp0port0." Refer to the Programmer's Guide for your particular protocol for a description of all the link configuration options.

Figure B-4. Session "icp0port0": Protocol-specific Parameters

```
  // -------------------------
  // FMP protocol-specific parameters.
  // -------------------------
  // DataRate = 9600;
  // ClockSource = "external";
  ClockSource = "internal";
  // NumLeadSync = 5;
  // Parity = "Even";
  // CharSet = "EbcdicCRC16";
  // TransBlkSize = 2048;
  // DataTranslation = "Table3";
  // BufferTimer = 2000;
  // ModemControl = "FDX1";
  ModemControl = "FDX2";
  // FeedID = 555;
  // MessageBlocking = "Disable";
  MessageBlocking = "DataBlk";         // best for switch task.
  // BlockChecking = "Disable";
  // QLimit = 1024;
  // ETBEnable = "Yes";
  // AsyncTermChar = 32;
  // NumTermChar = 1;
  // UsrDataRate = 2400;
  // ElecInterface   = "EIA232";
  // MsgBlkSize = 1024;
  WriteType = "Transparent";
}
```

Figure B-5 defines the protocol-independent parameters and the protocol-specific parameters for additional ICP0 sessions.

Figure B-5. Additional ICP0 Session Definitions

```
icp0port1
{  Protocol          = "FMP";
   Transport         = "Conn0";
   BoardNo           = 0;
   PortNo            = 1;
   AsyncIO           = "Yes";
   AlwaysQIO         = "Yes";
   Timeout           = 100;
   ModemControl      = "FDX2";
   ClockSource       = "internal";
   WriteType         = "Transparent";
   MessageBlocking   = "DataBlk";
}
icp0port2
{  Protocol          = "FMP";
   Transport         = "Conn0";
   BoardNo           = 0;
   PortNo            = 2;
   AsyncIO           = "Yes";
   AlwaysQIO         = "Yes";
   MessageBlocking   = "DataBlk";
   ClockSource       = "internal";
   ModemControl      = "FDX2";
}
icp0port3
{  Protocol          = "FMP";
   Transport         = "Conn0";
   BoardNo           = 0;
   PortNo            = 3;
   AsyncIO           = "Yes";
   AlwaysQIO         = "Yes";
   ModemControl      = "FDX2";
   ClockSource       = "internal";
   MessageBlocking   = "DataBlk";
}
icp0port4
{  Protocol          = "FMP";
   Transport         = "Conn0";
   BoardNo           = 0;
   PortNo            = 4;
   AsyncIO           = "Yes";
   AlwaysQIO         = "Yes";
   Timeout           = 100;
   ModemControl      = "FDX2";
   WriteType         = "Transparent";
   ClockSource       = "internal";
   MessageBlocking   = "DataBlk";
}
```

Figure B-6 defines the sessions for ICP1, which is an ICP_IP device implementing the "ipnull" interface. The definitions for icp1port0 through icp1port3 support the switch.cfg definitions for the apinull0, apinull1, apinull2, and apinull3 test programs (see Section A.8.1 through Section A.8.4).

Figure B-6. ICP1 Session Definitions (ICP_IP Device)

```
//-------------------------------------------------
// ipnull device.
// This is for use by an ICP_IP device implementing the "ipnull" interface.
//-------------------------------------------------
icp1port0
{  Protocol           = "FMP";          // FMP may be specified for "ipnull"
   Transport          = "Conn0";
   BoardNo            = 1;
   PortNo             = 0;
   Timeout            = 100;
   AsyncIO            = "Yes";
   AlwaysQIO          = "Yes";
}
icp1port1
{  Protocol           = "FMP";
   Transport          = "Conn0";
   BoardNo            = 1;
   PortNo             = 1;
   Timeout            = 100;
   AsyncIO            = "Yes";
   AlwaysQIO          = "Yes";
}
icp1port2
{ <similar definitions as other icp1 ports>
        .
        ..
}
icp1port3
{ <similar definitions as other icp1 ports>
        .
        ..
}
icp1port4
{ <similar definitions as other icp1 ports>
        .
        ..
}
```

Figure B-7 defines the sessions for ICP2, which is an ICP_IP device implementing the "ipnull" interface. This example also illustrates why you might increase the MaxOutQ parameter. ICP3 through ICP7 sessions are omitted from this example since they are similar to ICP1 and ICP2.

Figure B-7. ICP2 Session Definitions (ICP_IP Device)

```
//-----------------------------------------
// ICP 2 and ICP 3 are defined in bootcfg.sw as ICP_IP devices,
// and are loaded with the ipnull protocol.
//-----------------------------------------
icp2port0
{  Protocol          = "FMP";
   Transport         = "Conn0";
   BoardNo           = 2;
   PortNo            = 0;
   Timeout           = 100;
   AsyncIO           = "Yes";
   AlwaysQIO         = "Yes";
   MessageBlocking   = "DataBlk";
}
icp2port1
{  Protocol          = "FMP";
   Transport         = "Conn0";
   BoardNo           = 2;
   PortNo            = 1;
   Timeout           = 100;
   AsyncIO           = "Yes";
   AlwaysQIO         = "Yes";
   MessageBlocking   = "DataBlk";
       //-----------------------------------------
       // Increasing MaxOutQ from the default of 10 helps fmpalp to send
       // data at high data rates.  If you get errors similar to these
       // on the Freeway console:
       //
       //       process_reads: dlWrite failed (-11909)
       //       from icp5port1 to icp5port0.
       //
       // then it may be necessary to increase MaxOutQ further.
       //-----------------------------------------
   MaxOutQ           = 40;
}
icp2port2
{<icp2port2 through icp2port4 have similar definitions as other icp2 ports>
       .
       ..
}
```

B.3. TSI Configuration File (swtcfg)

Figure B-8 is an example TSI configuration file (`swtcfg`) for the Message Switch. The `swtcfg` file is used in the procedures described in Chapter 2.

Figure B-8. TSI Configuration File for Message Switch

```
//--------------------------------------------------
//
// swtcfg
//
// Define a main section and a section for a shared memory connection.  The
// defaults of most of the parameters are used, so they do not show up in this
// file.  See the TSI Reference Guide for a description of all TSI
// configuration parameters.
//
// -------------------------------------------------
// "main" section.  When present, the main section must be the first section
// defined in the DLI configuration file.
// -------------------------------------------------

main
{ AsyncIO    = "Yes";           // SRAs must use asynchronous I/O.
  MaxBuffers = 450;             // Default of 1024 uses too much RAM.
  MaxBufSize = 1300;            // 1224 data bytes plus 76 bytes DLI overhead.
}


// -----------------------------------------
// Definition for a shared-memory transport interface with the msgmux.
// -----------------------------------------

Conn0
{ Transport   = "shared-memory";  // Shared memory transport mechanism.
  AsyncIO     = "Yes";            // SRAs must use asynchronous I/O.
  Server      = "freeway_0";      // Task name of peer server task.  Matches
                                  //    ServerName parameter in muxcfg.sra.
  ShmPeerName = "Server2";        // Name of peer connection definition in
                                  //    muxcfg.sra.
}
```

Index

Customer Report Form

Customer Report Form

This document was produced from DocBook SGML source, using the DocBook 3.1 DSSSL stylesheets.

We at Protogate are constantly striving to improve our products. If you have any suggestions or problems you would like to report regarding our hardware, software, or documentation, please complete the following form and mail it to us at Protogate, Inc., 12225 World Trade Drive, Suite R, San Diego, CA, 92128, USA. Or contact us via email: <sales@protogate.com>, voice: (858) 451-0865, or fax: (877) 473-0190. Please also include the document title or number and the section and page number, if applicable.

Your Name and Phone Number:

_____

Company:

_____

Address:

_____

_____

_____

Product:

_____

Problem or Suggestion:

_____

_____

_____

_____

_____

Thank you.