Introduction

The InnoSys Airline Terminal Emulator ("IATE") Sun IATE Gateway allows a SPARCstation running the Solaris or SunOS operating system to connect a diverse range of devices to an airline ALC or X.25 host. The Sun IATE Gateway is one of a line of products from InnoSys Incorporated that includes terminal emulators, gateways, API's, TCP/IP-->ALC conversion devices, and protocol converters.

This document describes the installation procedure and the configuration options for the Sun IATE Gateway. Both the X.25 and the ALC installation procedures are included in this document. The procedures for running the gateway and diagnostic programs are also described. The Sun IATE Gateway includes:

- gateway software
- diagnostic software
- INSCC-S communications card

The document is written for use with Sun O.S. 4.x and Solaris 2.x.

In order to complete this installation, super-user privileges on the target system (that is, the ability to log in as "root") are required. The term "edit" is used in this document to mean using a text editor such as **vi** or **ex** to change a text file. Any text editor may be used.

System Requirements

The Sun workstations which are recommended for use with the InnoSys IATE Sun gateway product include the:

- SPARC Classic
- SPARC 4
- SPARC 5
- SPARC 10
- SPARC 20
- Ultra 1
- Ultra 2

The workstation should be configured with both a CD drive and a floppy drive. A Sun monochrome or color Monitor is recommended. While an Ascii terminal can be used instead of a regular Sun monitor, most customers who try this find it unsatisfactory. 32 Meg of RAM and a 1-gigabyte hard disk are sufficient. Solaris version 2.5.1 or 2.6 is recommended.

ALC and X.25 Connections

The Sun IATE package contains two gateways: one is the IATE ALC Gateway and the other is the X.25 Gateway. The ALC Gateway handles connections to ALC hosts and to terminal clients. The X.25 Gateway handles connections to X.25 hosts. The ALC Gateway may be configured to use the host connection(s) provided either by the X.25 Gateway or by direct connection to an ALC host.

On an ALC data line, only the IATE ALC Gateway is used.

On an X.25 data line, the IATE ALC Gateway is used as a client of the X.25 Gateway.

NOTE: The ALC Gateway is used with both an ALC data line and an X.25 data line. The ALC configuration file entries will depend on which type of line is being configured. Refer to the “ALC Gateway Configuration” section of this manual for further details.

Preparation

Before installing the INSCC-S card and Sun IATE Gateway software, set up the target SPARCstation, LAN, modem and printers. Also, obtain the Interchange Address (IA) and Terminal Addresses (TAs) assigned to this gateway at the airline host. In addition, when configuring the X.25 gateway, obtain the X.121 remote and local addresses, the range of logical channel numbers for SVCs or PVCs, the Network User ID, and any other relevant X.25 configuration information required by the airline host or network.

Refer to the owner's manuals provided by Sun for details on setting up the SPARCstation that will be used as the gateway.

Because the Sun IATE Gateway runs on a TCP/IP LAN, it is important to make sure that the network card and the TCP/IP protocol stack are properly installed, configured, and operational before proceeding with the gateway installation process. Check that this SPARCstation can be "pinged" from another workstation before continuing.

Once the SPARCstation and LAN are ready, verify that the airline modem is properly installed and communicating with the airline host. The modem should be cabled to the INSCC-S card after the card has been installed in the SPARCstation.

Getting the Gateway Running

There are five steps to take to get your NT Gateway up and running:

1) Check that the Sun hardware is properly set up and that TCP/IP protocol stack is correctly installed. These steps are described in more detail in the “System Requirements” and “Preparation” sections of this manual. Use the “Ping” command to verify that TCP/IP is properly installed. **IF YOU CANNOT PING THE GATEWAY HARDWARE SUCCESSFULLY, THE Sun IATE GATEWAY WILL NOT RUN**.

2) Install the INSCC-s card and then install the gateway software. These steps are described in more detail in the “Installation” section of this manual.

3) Configure the gateway. The easiest way to get started is to edit one of the sample configuration files in the iate_product/alc_gate/ directory. If the name of the configuration file used is not “scfg”, then the script for running the gateway must be changed to point to this configuration file. All of the gateway configuration options are described in more detail in the “Configuration” section of this manual.

4) To run the gateway, use one of the scripts in the “Starting a Gateway or Printer” section of this document, or issue the commands manually from the command prompt.

5) Check the output of the gateway after it starts to see if the gateway is running properly. Appendices E and F provide examples of the diagnostic output of the gateway when it starts up successfully.

«««

Installation

Installation of the Sun IATE requires the following steps.

- Installing the S-bus card
- Transferring files to the machine
- Installing the IALC device driver, and making special device nodes
- Making the Gateway known to UNIX
- Configuring the Gateway
- Configuring printers
- Distributing and installing fonts
- Distributing user files

Each of these steps is described in this section. Note that the software installation and configuration procedures differ between Solaris and Sun O.S.

Installing the S-Bus Card

Follow these steps to install the INSCC-S card into an available S-bus slot in the target SPARC computer system:

1) Halt the machine and turn it off.

2) Remove the cover from the machine.

3) Remove the face plate from an available S-bus slot.

4) Install the INSCC-S card in the S-bus slot . (Refer to SPARCstation installation documentation for details.)

5) Re-assemble the machine and plug it back in.

6) Re-boot the machine.

Transferring Files to the Machine

Each step of the installation should be performed in the order given here. It is necessary to log on as **root** to perform these operations.

Installation for Solaris 2.x:

Step 1: Stop the volume manager

If the volume management daemon (**vold**) is running it will probably need to be disabled for the floppy driver to work correctly. Check to see if **vold** is running by entering:

```
ps -ae | grep vold
```

If **vold** is running, use **volmgt** to shut it down, as follows:

```
/etc/init.d/volmgt stop
```

Step 2: Retrieve the installation script

Insert the disk labeled **license/installation** into the floppy drive. Then enter the following command:

```
tar xvf /dev/rdiskette extract
```

Step 3: Run the installation script

The following command will create an **iate_product** subdirectory under the current directory, and will install the product there.

```
./extract
```

Alternatively, provide the installation script with the full path name of the directory into which the product should be installed. For example:

```
./extract /usr/iate_product
```

The installation disks may be read in any order. The system utility "whoami" needs to be found by ./extract. When the installation is complete, enter "done" to exit the installation script.

Step 4: Restart the volume manager

At this point, **vold** can be turned back on to restore its automatic mounting capabilities or to support the OpenWindows File Manager. To reactivate **vold**, enter:

```
/etc/init.d/volmgt start
```

Installation for Sun O.S. 4.x:

Step 1: Retrieve the installation script

Insert the disk labeled **license/installation** into the floppy drive. Then enter the following command:

```
tar xvf /dev/rfd0c extract
```

Step 2: Run the installation script

The following command will create an **iate_product** subdirectory under the current directory, and will install the product there.

```
./extract
```

Alternatively, provide the installation script with the full path name of the directory into which the product should be installed. For example:

```
   ./extract /usr/iate_product
```

Installing the INSCC Device Driver and Making Special Device Nodes

The S-bus device driver is an installable device driver. This means that it is not necessary to rebuild the UNIX kernel to install the device driver. However, on a Sun O.S. 4.x system, the device driver will need to be re-installed whenever the system is booted.

It is necessary to be logged in as **root** in order to perform the following commands to install the device driver.

Device Driver Installation for Solaris 2.x:

Step 1: Ensure that the INSCC-S card is installed.

If the INSCC-S card has not already been installed as described earlier, power down the machine, install the card, and then reboot the machine.

Step 2: Add a device specification to the system's device table.

Add the following line to /etc/devlink.tab. If the line already appears, don't add it a second time. `<tab>` means to enter a Tab character using the tab key. The final character is a zero, not a capital letter "O".

```
      type=ddi_network;name=INNO,insccs<tab>\M0
```

Step 3: Load the device driver software.

Go to the device sub-directory of the installation directory. For example, if the IATE product files were extracted into the **/home/iate_product** directory, then the device directory would be **/home/iate_product/device**:

```
           cd /home/iate_product/device
```

Enter the following command:

```
      ./loaddev
```

The **loaddev** script loads the loadable device driver and calls **modstat** to display the status of loaded kernel modules.

Step 4: Determine which device to use.

The utility **show_innosb** can be used to determine the device names which will be available for use on the system. Most of the provided utility programs and scripts assume (or should assume) that the device is **/dev/innosb0**. If the programs or scripts cannot find /dev/innosb0, run **show_innosb** to find out which devices are really available. Then change the shell scripts and configure the Gateways to use the appropriate device. In particular, it is important to make sure that the **loadalc** script (in the sub-directory **iate_product/alc_line/**) specifies the device name correctly.

If **/dev/innosb0** is not being used, it is necessary to supply a `BOARD_NUMBER` to the ALC Gateway. For these purposes, use the number appended to “innosb” in the device name as the board number.

If an X.25 line is being used, it is necessary to specify the device name to the X.25 Gateway. To do this, use the `-b` command line option when starting the X.25 Gateway. For example, the following command starts the X.25 Gateway using the device **/dev/innosb1**:

```
   x25gate -b/dev/innosb1
```

Device Driver Installation for Sun O.S. 4.x:

Step 1: Ensure that the INSCC-S card is installed.

If the INSCC-S card has not already been installed as described earlier, power down the machine now, install the card, and then reboot the machine.

Step 2: Load the device driver software.

Go to the device sub-directory of the installation directory. For example, if the IATE product files were extracted into the **/home/iate_product** directory, then the device directory would be **/home/iate_product/device**:

```
          cd /home/iate_product/device
```

Enter the following command:

```
   ./loaddev
```

The **loaddev** script loads the loadable device driver and calls **modstat** to display the status of loaded kernel modules. For a Sun O.S. 4.x installation, something similar to the following should be displayed:

```
   Id  Type  Loadaddr  Size  B-major  C-major  Sysnum  Mod Name

   1   Drv   ff14b000  2000             59.            insccs
```

Step 3: Create new device nodes.

The first time that the driver is loaded, the **mknod** command should be used to make a special device file for the **insccs** driver. Make a note of the *C-major number* given in the output of the **modstat** command. The C-major number is the major device number, which will be used with the **mknod** command. In the example above, there is just one device, with major device number 59, and device ID 1. (These are only sample values; the target system's values may be different) The ID number 1 corresponds to board number 0. (Board numbers start from zero.) The appropriate **mknod** command for this example device would be:

```
   mknod /dev/innosb0 c 59 0
```

Issue a **mknod** with the major device number appropriate for the target system.

If there is more than one INSCC board in the SPARCstation, a separate **mknod** must be issued for each board. Use the board number as the minor device number. Name the device by appending the board number to "innosb". A second board would be #1 (since board numbers start from 0). Continuing from the example above, an appropriate **mknod** command for a second board would be:

```
   mknod /dev/innosb1 c 59 1
```

Step 4: Determine which device to use.

Most of the provided utility programs and scripts assume (or should assume) that the device is **/dev/innosb0**. If **/dev/innosb0** is not available, change the shell scripts and configure the Gateways to use the appropriate device. In particular, it is important to make sure that the **loadalc** script (in the sub-directory **iate_product/alc_line/**) specifies the device name correctly.

If **/dev/innosb0** is *not* being used, a `BOARD_NUMBER` must be supplied to the ALC Gateway. For these purposes, use the number appended to "innosb" in the device name as the board number.

If an X.25 line is being used, it is necessary to specify the device name to the X.25 Gateway. To do this, use the `-b` command line option when starting the X.25 Gateway. For example, the following command starts the X.25 Gateway using the device **/dev/innosb1**:

```
   x25gate -b/dev/innosb1
```

Note:
If the system configuration is changed to add other loadable device drivers, it may be necessary to create special device file(s) using a different major device number. If so, be sure to remove the previous special device file(s) for the INSCC-S device. For example, to remove **/dev/innosb0**:

```
   rm /dev/innosb0
```

Making the ALC Gateway Known to the Network

In order to make the IATE ALC Gateway server known to the IP network, add the following line to the **/etc/services** file:

```
   ialcserver  1413/tcp
```

The default service name that the ALC Gateway uses is "ialcserver". Use lower case. (Case is significant in recent versions of Solaris.) If a different Gateway name will be used, the Gateway configuration must be changed as well. See the Gateway Configuration section, below. The officially registered port number is 1413. It is OK to use another port number that is available (not in use by any other service) on the system. Do not use port numbers below 1024 (they are reserved). It is necessary to use “/tcp” in the entry, as TCP/IP is the protocol that the Gateway uses to communicate with terminals and printers on the network.

For additional information on installing services, enter the following command to see the online manual entry for services:

```
   man services
```

Making the X.25 Gateway Known to the Network

Perform the following procedure only if the gateway will communicate with the airline host through an X.25 network. In this case, both the ALC and X.25 Gateways are used. The ALC Gateway communicates with the airline host(s) through the X.25 Gateway.

In order to make the IATE X.25 Gateway server known to the IP network, add the following line to the **/etc/services** file:

```
   x25gate    1412/tcp
```

The default service name that the X.25 Gateway uses is “x25gate”. Use lower case. (Case is significant in recent versions of Solaris.) The officially registered port number is 1412; but, if necessary, it is OK to use another port number that is available (not in use by any other service) on the system. Do not use port numbers below 1024 (they are reserved). It is important to use “/tcp” in the entry, as TCP/IP is the protocol that the X.25 Gateway uses to communicate with the ALC Gateway.

Gateway Configuration

The ALC and X.25 Gateways offer the user a number of configuration options. The options are separated into a number of groups. Below is a list of the the different groups of gateway options with the page number each group is found on, followed by a list of the individual configuration options (the ALC Gateway options are listed first, then the X.25 Gateway options) with the name of the group in which the option is found.

Group name	Page #
*BROADCAST TAS (ALC Gateway)	17
*CRT ANSWERBACKS (ALC Gateway)	16
*GATEWAY DEFAULTS (ALC Gateway)	13
*HOST CONNECTION (ALC Gateway)	20
*HOST TYPE (ALC Gateway)	13
*IAS (ALC Gateway)	16
*LINE NUMBERS (ALC Gateway)	17
*OBJECT CONNECTION DEFAULTS (ALC Gateway)	18
*OBJECT DEFINITIONS (ALC Gateway)	19
*PRINTER ANSWERBACKS (ALC Gateway)	15
*PROTOCOL DESCRIPTION (ALC Gateway)	14
*SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)	24
*SWITCH (X.25 Gateway)	23
*TWO-VALUE CONFIGURATION ITEM (X.25 Gateway)	25

Option name	Group Name
ACTIVITY_TIMER	*HOST CONNECTION (ALC Gateway)
API_THROTTLE_INTERVAL	*OBJECT CONNECTION DEFAULTS (ALC Gateway)
AUTO_ANSWER	*PROTOCOL DESCRIPTION (ALC Gateway)
BOARD_NUMBER	*HOST CONNECTION (ALC Gateway)
CTS_REQUIRED	*PROTOCOL DESCRIPTION (ALC Gateway)
CUD	*HOST CONNECTION (ALC Gateway)
DATA_IA	*IAS (ALC Gateway)
DCD_REQUIRED	*PROTOCOL DESCRIPTION (ALC Gateway)
DEFAULT_SEG_SIZE	*PROTOCOL DESCRIPTION (ALC Gateway)
DONT_FILTER_DATA_TO_HOST	*GATEWAY DEFAULTS (ALC Gateway)
DSR_REQUIRED	*PROTOCOL DESCRIPTION (ALC Gateway)
ENABLE_ENCODING	*GATEWAY DEFAULTS (ALC Gateway)
EOMC	*PRINTER ANSWERBACKS & *CRT ANSWERBACKS (ALC Gateway)
EOMI	*PRINTER ANSWERBACKS & *CRT ANSWERBACKS (ALC Gateway)
EOMPB	*PRINTER ANSWERBACKS & *CRT ANSWERBACKS (ALC Gateway)
EOMU	*PRINTER ANSWERBACKS & *CRT ANSWERBACKS (ALC Gateway)
EXTENDED_CHARS	*PROTOCOL DESCRIPTION (ALC Gateway)
FREEZE_INITIAL_ADDRESS	*HOST CONNECTION (ALC Gateway)
HEARTBEAT_REQUIRED	*OBJECT CONNECTION DEFAULTS (ALC Gateway)
IA_NATIVE_8_BIT	HOST CONNECTION (ALC Gateway)
INET_ADDRESS	*GATEWAY DEFAULTS (ALC Gateway)
LIMIT_SEGS_PER_POLL	*PROTOCOL DESCRIPTION (ALC Gateway)
LOCAL_ADDRESS	*HOST CONNECTION (ALC Gateway)
MAX_SEGS_PER_POLL	*PROTOCOL DESCRIPTION (ALC Gateway)
NET_HOST_NAME	*GATEWAY DEFAULTS (ALC Gateway)
NOTIFICATION_DELAY	*PROTOCOL DESCRIPTION (ALC Gateway)
NO_INITIAL_CALL	*HOST CONNECTION (ALC Gateway)
OVERRIDE_DEFAULT_PVC_IA0	*HOST CONNECTION (ALC Gateway)
PAD_TYPE	*HOST CONNECTION (ALC Gateway)
POLLING_IA	*IAS (ALC Gateway)
PORT_NAME	*HOST CONNECTION (ALC Gateway)

PORT_NAME	*HOST CONNECTION (ALC Gateway)
PORT_NUMBER	*HOST CONNECTION (ALC Gateway)
PVC_LCN	*HOST CONNECTION (ALC Gateway)
QUEUE_SLACK	*GATEWAY DEFAULTS (ALC Gateway)
QUEUEITEM_SLACK	*GATEWAY DEFAULTS (ALC Gateway)
REMOTE_ADDRESS	*HOST CONNECTION (ALC Gateway)
SEGMENTATION	*PROTOCOL DESCRIPTION (ALC Gateway)
SERVER_NAME	*GATEWAY DEFAULTS (ALC Gateway)
SYNCS_BETWEEN	*PROTOCOL DESCRIPTION (ALC Gateway)
TA_POLLING_ENABLED	*PROTOCOL DESCRIPTION (ALC Gateway)
TA_TIMEOUT	*OBJECT CONNECTION DEFAULTS (ALC Gateway)
TXWAITCOUNT	*PROTOCOL DESCRIPTION (ALC Gateway)
USER_ID	*HOST CONNECTION (ALC Gateway)
X25_GATEWAY	*HOST CONNECTION (ALC Gateway)

CTS	SWITCH (X.25 Gateway)
DCD	SWITCH (X.25 Gateway)
DCE	SWITCH (X.25 Gateway)
DSR	SWITCH (X.25 Gateway)
DTE	SWITCH (X.25 Gateway)
EXTENDED_LAPB	SWITCH (X.25 Gateway)
EXTENDED_PACKET	SWITCH (X.25 Gateway)
FLAG	SWITCH (X.25 Gateway)
INTERNAL_CLOCK	SWITCH (X.25 Gateway)
K	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
LINE_SPEED	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
N2	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
PACKET_SIZE	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
PACKET_SIZE_NEGOTIATION	SWITCH (X.25 Gateway)
PVC	TWO-VALUE CONFIGURATION ITEM (X.25 Gateway)
SERVICE	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
START_SABM	SWITCH (X.25 Gateway)
SVC	TWO-VALUE CONFIGURATION ITEM (X.25 Gateway)
T10	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
T11	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
T12	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
T13	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
T1	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
T24	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
T2	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
T3	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
T4	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
WINDOW_SIZE	SINGLE-VALUE CONFIGURATION ITEM (X.25 Gateway)
WINDOW_SIZE_NEGOTIATION	SWITCH (X.25 Gateway)

ALC Gateway Configuration

(*Note:* The ALC Gateway is used for both ALC and X.25 connections. Therefore the ALC Gateway must be configured regardless of which type of network will be used.)

The ALC Gateway configuration file is a normal text file, editable with any ASCII text editor (such as **vi** or **ex**). It contains several sections, each preceded by a keyword that names the section. Each section may contain one or more option specifications as described below. The keywords naming each section must start with an asterisk (*). Comment lines start with a pound sign (#); the gateway will ignore them. Sample Gateway configuration files, such as “scfg”, are provided in the iate_server installation directory.

Listed below are the section keywords and options that are available for each keyword. **ALL PARAMETERS AND OPTIONS ARE CASE-SENSITIVE!**

*GATEWAY DEFAULTS

In the Gateway Defaults section of the configuration file, any option not specifically supplied will revert to its default value, as shown in the table below.

Option	Argument	Default
NET_HOST_NAME	*name*	local host name
SERVER_NAME	*name*	ialcserver
INET_ADDRESS	*address*	INADDR_ANY
ENABLE_ENCODING	*none*	Encoding not used
QUEUE_SLACK	*number*	0
QUEUEITEM_SLACK	*number*	0
DONT_FILTER_DATA_TO_HOST		not enabled

NET_HOST_NAME is not normally used since the default behavior, using the local host's name, is usually appropriate. NET_HOST_NAME is only used when the network address family that will be used to establish connections to the configured objects is not the same as the network address family of the gateway machine. In this case, use NET_HOST_NAME to specify the host name of a machine which has the proper network address family.

SERVER_NAME specifies the service name from that the gateway will use; normally this need not be specified, as the default service name (ialcserver) is usually correct.

INET_ADDRESS specifies the IP address on which the ALC Gateway should listen for incoming object connections. Normally this option is not specified. It is primarily used on systems with multiple Ethernet interfaces (with different host addresses), where a separate Gateway serves each interface.

ENABLE_ENCODING causes the data that passes between the gateway and workstation clients to be encoded. This prevents plain text from being passed over the TCP/IP network. Encoding is only used with each individual client if the version of the api that the client is using supports encoding. See Appendix I for "Frequently Asked Questions about Encoding".

QUEUEITEM_SLACK and QUEUE_SLACK are only necessary when shared TA's are being used. These parameters increase the number of buffers that the gateway allocates. Programmers working with shared TA's should check with InnoSys to see if either of these parameters should be used. When used, typical values are: QUEUEITEM_SLACK = number of shared TA applications running * 2; QUEUE_SLACK = number of shared TA applications running * 9.

DONT_FILTER_DATA_TO_HOST is relevant for the PARS host type only. The default behavior of the gateway is to remove the following characters out of messages are going to the host.

alc 0x0e	ascii 0x3d	=
alc 0x2b	ascii 0x3c	<
alc 0x2c	ascii 0x2b	+
alc 0x3a	ascii 0x3f	?
alc 0x3c	ascii 0x25	%

*HOST TYPE

The HOST TYPE section should contain a single line specifying the type of airline host system to which the Gateway should connect. For example, for a SABRE connection, this section should contain the line:

```
SABRE
```

CAUTION: The HOST TYPE entry must **not** have any trailing spaces because they will be interpreted as part of the host name.

The complete set of valid host type names is:

SABRE, PARS (used for Worldspan), APOLLO, DATAS, SODA, SHARES, UNIPARS, KLM, UNISCOPE, CODACOM, AC100, JAL, ANA, EGYPT, ABACUS, ISEA, AMADEUS, GALILEO, CPARS, SWISSAIR, SITACARGO, BABS, OLYMPIC, KOREAN, and SINGAPORE.

Note: Air New Zealand/Carina hosts require PARS in the HOST TYPE section, and PAD type AIRNZ in the X.25 HOST CONNECTION section (described later).

*PROTOCOL DESCRIPTION

The following options are available in the Protocol Description section of the configuration file. If an option is not listed the default setting will be used.

Several options are two-state "toggles", plus ('+') to enable the option or minus ('-') to disable the option. If a toggle option's default value is '+' it is enabled by default. If the option's default is '-' it is disabled by default. For this type of option, the '+' or '-' character should immediately precede the option name, without intervening spaces. For example:

```
-DCD_REQUIRED
+LIMIT_SEGS_PER_POLL
```

The DEFAULT_SEG_SIZE, MAX_SEGS_PER_POLL, and TXWAITCOUNT options take a numeric argument. For these options, specify the option name, followed by a blank space and a numeric argument. For example, the following line will set DEFAULT_SEG_SIZE to 960:

```
DEFAULT_SEG_SIZE 960
```

Option	Argument	Default value
NOTIFICATION_DELAY	*number*	3
EXTENDED_CHARS	+/-	depends on host type
TA_POLLING_ENABLED	+/-	-
DSR_REQUIRED	+/-	+
DCD_REQUIRED	+/-	+
CTS_REQUIRED	+/-	+
SEGMENTATION	+/-	depends on host type
LIMIT_SEGS_PER_POLL	+/-	+
SYNCS_BETWEEN	+/-	-
AUTO_ANSWER	+/-	+
DEFAULT_SEG_SIZE	*number*	98
MAX_SEGS_PER_POLL	*number*	30
TXWAITCOUNT	*number*	0

NOTIFICATION_DELAY specifies how long after it first detects the host line is down that the onboard code should wait before informing the gateway.

EXTENDED_CHARS specifies whether or not the ALC Gateway should support the escape sequences associated with the SABRE extended character set. The default value depends on the host type. By default, extended character support is enabled for KOREAN and SABRE hosts and disabled for all other hosts.

TA_POLLING_ENABLED specifies whether or not the ALC Gateway and INSCC onboard software should support "TA Polling" on a SABRE line. Leave this option disabled unless otherwise instructed by the CRS. TA Polling is used in some instances where an IA has not been dedicated to a terminal cluster.

DSR_REQUIRED, DCD_REQUIRED, and CTS_REQUIRED specify whether the indicated signals are required. When a signal is required, it must be active in order for the INSCC onboard software to communicate with the airline host. When a signal is not required, the INSCC onboard software will ignore it and will attempt communications regardless of the signal's status. These three options apply only to ALC connections, not X.25 connections. For equivalent X.25 configuration options, **see the X.25 Gateway Configuration** section.

SEGMENTATION controls whether or not messages are broken into segments for transmission to the host. DEFAULT_SEG_SIZE specifies the size, in bytes, for data message segments sent to the host. The INSCC onboard software will ensure that each data segment sent is no longer than the indicated size. (This only affects segments sent *to* the host. The ALC Gateway and onboard software can still accept longer segments *from* the host.)

LIMIT_SEGS_PER_POLL specifies whether or not the INSCC onboard software should impose a limit on the maximum number of data message segments that it will send in response to a single poll. If this option is set to '+', then MAX_SEGS_PER_POLL can be used to specify the maximum number of segments sent for each poll. If LIMIT_SEGS_PER_POLL is set to '+' but MAX_SEGS_PER_POLL is not specified, then the limit will default to 30 segments per poll.

SYNCS_BETWEEN specifies whether or not the INSCC onboard software should insert an ALC synchronization sequence (S1 S2) between successive segments sent to the host in response to a single poll. This option applies only to ALC connections, not X.25 connections.

TXWAITCOUNT specifies a transmission wait count. For some hosts, this option is required in order to regulate the response rate. Do not use this option unless it is suggested by InnoSys or the airline host.

The AUTO_ANSWER option works in conjunction with the PRINTER ANSWERBACKS and CRT ANSWERBACKS sections described below. This option specifies whether or not the gateway should send an automatic acknowledgment for each data segment received from the host. If this option is enabled, the gateway will acknowledge each received segment that requires acknowledgment immediately after forwarding it to a destination terminal/printer or application program. This means that the gateway will not wait for acknowledgment from the client, but will instead generate an acknowledgment on its own. In general, this option is not recommended. It must not be used on protected TA objects, where a printer or application should generate end-to-end acknowledgement for each protected segment.

*PRINTER ANSWERBACKS

The PRINTER ANSWERBACKS section may contain up to four lines, each specifying a type of EOM character which requires the ALC Gateway to send a printer answerback message. When the Gateway sends a printer data message segment to a printer client, if the segment has one of the specified EOM characters, the Gateway will send an answerback (automatically if auto-answer is enabled, or else after receiving acknowledgment from the printer client). One or more of the following EOM types may be specified, one per line. The default setting is `EOMI` and `EOMU`.

```
EOMC
EOMI
EOMU
EOMPB
```

*CRT ANSWERBACKS

The `CRT ANSWERBACKS` section may contain up to four lines, each specifying a type of EOM character which requires the ALC Gateway to send a CRT answerback message. When the ALC Gateway sends a CRT data message segment to a terminal client, if the segment has one of the specified EOM characters the Gateway will send an answerback (automatically if auto-answer is enabled, or else after receiving acknowledgment from the terminal client). One or more of the following EOM types may be specified, one per line. CRT answerbacks are not normally used. When this section is present, the default setting is `EOMI` and `EOMU`.

```
EOMC
EOMI
EOMU
EOMPB
```

*IAS

The contents of the `IAS` section differs depending on whether a direct ALC connection or an X.25 connection is being configured.

For ALC connections, the `IAS` section includes the following two options:

Option	Parameter	Comments
`DATA_IA`	Data IA (ALC hexadecimal)	At least 1 `DATA_IA` option line is required. Up to 38 additional Data IAs are allowed.
`POLLING_IA`	Polling IA (ALC hexadecimal)	Optional.

Up to 39 Data IAs can be specified per SPARCstation. No more than one Polling IA may be specified. Following is an example of three `DATA_IA` option lines specifying Data IAs 01, 02, and 03, and Polling IA 01 on an ALC line. In this case, the leading zeros are optional.

```
DATA_IA 01
DATA_IA 02
DATA_IA 03
POLLING_IA 01
```

If a `POLLING_IA` is specified, it must be one of the `DATA_IA` values, and the `DATA_IA` entry must precede the `POLLING_IA` entry. If a `POLLING_IA` is specified, only polls to the `POLLING_IA` will be answered. *Note:* If `POLLING_IA` is used, the airline host must be specifically configured to support it. Not all hosts can support a `POLLING_IA`. The IA in each inbound data message (to the host) will be the IA associated with the object which generated the message.

If no POLLING_IA is specified, the ALC Gateway will answer polls for each DATA_IA as it is polled. A poll to any DATA_IA will result in messages from objects associated with that IA being sent to the host.

For X.25 connections, the format of the IAS section varies by PAD type. For the Air New Zealand and SABRE PAD types, each DATA_IA must be specified as a 4-character LNIA (Line and IA values, without intervening spaces). For example, to specify Line 01 and IA 02:

```
DATA_IA 0102
```

In this case, a leading zero *is required* for any line or IA value having just one nonzero digit, because the complete LNIA parameter value for X.25 must be exactly 4 hexadecimal digits long.

For the Apollo, Galileo, and Worldspan PAD types, each DATA_IA must be specified as a 1- or 2-character IA For example, to specify IA 07:

```
DATA_IA 7
```

POLLING_IA is NOT used with X.25.

*BROADCAST TAS

The optional BROADCAST TAS section allows the definition of a Broadcast TA for each Data IA on an ALC or X.25 line. Since this section includes *only* Broadcast TA definitions, there is no need for a label on each line. For each broadcast TA, enter a pair of values specifying the Data IA and the Broadcast TA — one pair per line.

Parameters

Data IA (ALC hexadecimal) Broadcast TA (ALC hexadecimal)

For example, to define Broadcast TA 08 on IA 01:

```
01 08
```

Do not define more than one Broadcast TA for any given Data IA. A Broadcast TA option line does not *define* a Data IA — it *refers* to a Data IA and defines a Broadcast TA to be associated with it. A DATA_IA line for each Data IA is still required.

*LINE NUMBERS

The optional LINE NUMBERS section sets a line number for use in addressing data messages over an X.25 network. The LINE NUMBERS section is only used with the Apollo, Galileo, and Worldspan PAD types. As in the BROADCAST TAS section, only the parameter values, one pair per line, without keywords are specified:

Parameters

Data IA (ALC hexadecimal) Line Number (ALC hexadecimal)

For each specified Data IA and Line Number pair, the ALC Gateway will place the specified Line Number into the first byte of the address sequence that begins each data message being sent for the corresponding Data IA, before forwarding the message to the X.25 Gateway.

For example, setting a Line Number of 92 for IA 1C would look like this:

```
1C 92
```

*OBJECT CONNECTION DEFAULTS

The `OBJECT CONNECTION DEFAULTS` section specifies three parameters which affect communications between the ALC Gateway and each attached client. These are called "Connection Defaults" because, although they take effect immediately after establishment of each object connection, an attached object can change them (through IATE API function calls).

The format of these parameters is similar to those given in the PROTOCOL DESCRIPTION section. For HEARTBEAT_REQUIRED, prefix the option name with a '+' or '-' character; and for the other arguments, append a blank space and the numeric argument. For example:

```
+HEARTBEAT_REQUIRED
TA_TIMEOUT  0
```

Any option not specifically supplied uses the default value shown in the table below.

Option	Arguments	Default value
HEARTBEAT_REQUIRED	+/-	-
TA_TIMEOUT	*number*	5 minutes (use 0 to disable)
API_THROTTLE_INTERVAL	*number*	1 second

If `HEARTBEAT_REQUIRED` is set to '+', it instructs the ALC Gateway to expect "heartbeat" or data messages from each attached client at least once every 60 seconds. If the ALC Gateway does not receive an expected heartbeat within 60 seconds, it will disconnect the object. This option overrides `TA_TIMEOUT` (described in the next paragraph), meaning that the heartbeat timeout is forced to 60 seconds *regardless* of any `TA_TIMEOUT` option setting that may be specified in the configuration file.

`TA_TIMEOUT` is used to set the amount of time after which the gateway will disconnect an object unless the gateway receives a message to or for that object. If `HEARTBEAT_REQUIRED` is not set, or if it is set to '-', but `TA_TIMEOUT` is set to a value greater than zero, then the ALC Gateway will expect heartbeats or data within the specified time period. However, in some cases, a client can send a "Reset 1 Minute" message (for more information, see the IATE API manual) to request that the ALC Gateway wait up to a full minute for the next heartbeat, before disconnecting the object.

If `HEARTBEAT_REQUIRED` is not set, or if it is set to '-', and `TA_TIMEOUT` is set to 0, then the ALC Gateway will not require heartbeats from any attached clients (in other words, it will not disconnect any object due to a lack of heartbeats).

The purpose of `HEARTBEAT_REQUIRED` is to protect against idle applications keeping TAs occupied. In contrast, the purpose of `TA_TIMEOUT` is to detect a "crashed" application's failure to disconnect from the gateway.

The API_THROTTLE_INTERVAL specifies the minimum interval between data transmissions from the API to the Gateway (over the API<-->Gateway TCP connection).

*OBJECT DEFINITIONS

There are no default values for the OBJECT DEFINITIONS section. This section is required. At least one object must be defined.

Aside from the three exceptions shown below, each object definition consists of one line of the form:

```
ia ta  type  object_name  group_name
```

- Exception #1, when using the Air New Zealand X.25 PAD type:

  ```
  ta  type  object_name  group_name
  ```

- Exception #2, when using the Apollo X.25 PAD type:

  ```
  ia ta  gtid  type  object_name  group_name
  ```

- Exception #3, when using the SABRE X.25 PAD type:

  ```
  lineiata  type  object_name  group_name
  ```

"ia" is the interchange address that is associated with this object.

"ta" is the terminal address that is associated with this object.

"gtid" is the 8-character global terminal identifier assigned by Apollo that is associated with this object. Usually only the last two digits of the gtid change for each object in the configuration file; the first six digits normally stay the same.

"lineiata" is the 6-character line number + interchange address + terminal address associated with this object:

The "type" must be one of the following:

```
TERMINAL
PRINTER
TERMINAL_API
PRINTER_API
```

The TERMINAL_API and PRINTER_API object types support dynamic linking from programs using the IATE API. For example, the API function APILinkToDyCrt will only connect to TERMINAL_API objects, whereas APILinkToName will connect to any available TERMINAL or TERMINAL_API object. Refer to the API manual for more information.

The object name is an ASCII string. It cannot have any imbedded blanks. Also, it should not have two consecutive asterisks "**" if the TA will be used as a member of a group for Dynamic TA linking. The object name is used to give a name to each ia-ta combination. Each object name in the configuration file must be unique.

The group name is an ASCII string with no blanks, or "**" if no group will be used. The group name is used to associate a number of TA's into one group addressable by a single name. The items in a group do not need to be next to each other in the configuration file.

Here is a sample from the **scfg** file shipped on the distribution media:

```
   1 04 TERMINAL      term04    **
   1 06 TERMINAL      term06    **
```

On IA 1, there are two terminal objects defined: for TA 4, term04; and for TA 6, term06. Neither of these TA's is a member of a group for dynamic linking

*HOST CONNECTION

The contents of the `HOST CONNECTION` section are different depending on whether an ALC connection or an X.25 connection is being configured.

For ALC connections, the `HOST CONNECTION` section includes these options:

Option	Argument	Default
BOARD_NUMBER	*board number*	0
PORT_NUMBER	*port number*	0
PORT_NAME	*port name*	(none)

The `BOARD_NUMBER` option is required. The *board number* is appended to the device driver base name, **innosb**, to get the device name which the ALC Gateway will use to communicate with the device driver, onboard software, and airline host. For example, if a line on board 0 is configured, the ALC Gateway will open **/dev/innosb0**; if a line on board 1 is being configured, the Gateway will open **/dev/innosb1**, etc.

The `PORT_NUMBER` option is required only when configuring a 2-port INSCC-S board for connection to 2 ALC lines. The two available port numbers are 0 and 1. (*Note:* The INSCC-S card has one physical connector. Dual-port operation is achieved through use of a a special adapter cable. The INSCC-S board's dual-port mode can only be used for ALC connections - when an INSCC-S board is connected to X.25, only one of its ports can be used. When using two ports for ALC, CTS is not supported on the second port. To make the second port run ALC properly, use the "-CTS" option, which means that CTS is ignored. Since loss of CTS on a constant carrier modem will not be noticed or reported, the only practical limitation of this is that switched carrier modems cannot be used.)

The `PORT_NAME` option is not required. If it is specified, the Gateway will include a port name in certain diagnostic messages. This can be helpful for diagnosing certain kinds of problems.

For X.25 connections, the `HOST CONNECTION` section includes these options:

Option	Parameter	Default
X25_GATEWAY	x25gate	This line is required
PORT_NAME	*port name*	Optional
PAD_TYPE	*PAD type name*	SABRE
PVC_LCN	*decimal number*	Required for PVCs
REMOTE_ADDRESS	*ASCII string*	Required for SVCs
LOCAL_ADDRESS	*ASCII string*	None

`OVERRIDE_DEFAULT_PVC_IA0`	*none*	
`FREEZE_INITIAL_ADDRESS`	*none*	
`IA_NATIVE_8_BIT`	*none*	
`USER_ID`	*ASCII string*	None
`CUD`	*ASCII string*	(See description below)
`NO_INITIAL_CALL`		
`ACTIVITY_TIMER`	*number of minutes*	10 (15 for Worldspan)

The `X25_GATEWAY` option is required. This is the name of the service over which the ALC Gateway and the X.25 Gateway communicate. This service must be defined in the "Services" file. (For an X.25 connection, the ALC Gateway does not communicate directly with the INSCC-S board. Instead, the ALC Gateway connects to the X.25 Gateway, which communicates with an INSCC-S board connected to the X.25 network.)

The `PORT_NAME` is optional. If it is specified, the Gateway will include a port name in certain diagnostic messages. This can be helpful for diagnosing certain kinds of problems. It is best to use different port names for ALC vs. X.25 connections.

The `PAD_TYPE` option specifies the type of X.25 PAD (Packet Assembler/Disassembler) with which the X.25 Gateway must communicate. The choices are: `AIRNZ (IATA PVC)`, `APOLLO`, `SABRE`, and `WORLDSPAN (IATA SVC)`. `SABRE` is the default. See Appendix G for more information.

Specify *either* the X.121 address(es) (the `REMOTE_ADDRESS` and, if necessary, the `LOCAL_ADDRESS`) *or* the `PVC_LCN` — this is how the gateway determines whether to set up an SVC or a PVC connection.

> The `PVC_LCN` option specifies a Permanent Virtual Circuit's Logical Channel Number (LCN). This option is required for PVC connections. (*Note:* In versions of the ALC Gateway prior to January 1995, the keyword `PVC_LCN` is not supported. Instead, use `DESTINATION_PVC`. Current versions of the ALC Gateway will accept either form.)
>
> The `REMOTE_ADDRESS` and `LOCAL_ADDRESS` options specify the X.121 Remote and Local addresses, respectively, for SVC connections. The maximum length of either field is 15 characters. The `REMOTE_ADDRESS` is required for SVC connections. If the X.25 network requires a DNIC, it should be placed at the beginning of the `REMOTE_ADDRESS` parameter value. (SABRE in Canada sometimes uses a DNIC of 3156. The combination of the DNIC and the `REMOTE_ADDRESS` is sometimes referred to as the "DNA".) Some airline hosts refer to the `REMOTE_ADDRESS` as the `"HUNT GROUP"`. For the Worldspan PAD type, one, two, or three `REMOTE_ADDRESS` entries can be used for an SVC connection. For all other PAD types, one `REMOTE_ADDRESS` entry is used for an SVC connection.
>
> The `LOCAL_ADDRESS` is optional, unless a Worldspan X.25 connection is being configured, or the network administrator requires a source address. (*Note:* In versions of the ALC Gateway prior to January 1995, the keywords `REMOTE_ADDRESS` and `LOCAL_ADDRESS` are not supported. Instead, use `DESTINATION_ADDRESS` and `SOURCE_ADDRESS`, respectively. Current versions of the ALC Gateway will accept both the old and new forms.)

Each ALC Gateway configuration file specifies just one PVC LCN or one SVC remote address. To configure a Gateway for more than one PVC LCN or SVC Remote Address, see the section below titled "Notes on Configuring Multiple IA's on a single X.25 Connection".

`OVERRIDE_DEFAULT_PVC_IA0` specifies whether or not the ALC Gateway should include the configured IA in messages to the host when using the AIRNZ PAD type. The Default behavior is to use an IA of 0 (zero) in messages to the host.

FREEZE_INITIAL_ADDRESS forces the gateway to ignore the calling address in the “Accept” packet. This option is only used with SABRE host types.

IA_NATIVE_8_BIT is used with the Worldspan PAD type to force the gateway to send and receive the IA in messages as two hex digits instead of as two reverse inverted PARS characters. This option is recommended for the Galileo host type.

The USER_ID specifies the *Network User ID (NUI)* optional user facility for SVC calls. This option is used only for X.25 SVC connections that require a NUI. Sometimes a Password is required in addition to a NUI. If so, it is usually included in the NUI field. For example, on a SABRE dial-up X.25 line, the NUI is concatenated onto the end of the Password (with no intervening spaces or delimiters) and the resulting string is entered into the USER_ID field.

The CUD specifies the *Call User Data* for SVC calls. This option is used only for X.25 SVC connections that require user data.

The default Call User Data for SABRE host connections is: c1 00 00 00

The default Call User Data for Worldspan host connections is: fe 12 10 00 a1 00 00 00 00 Worldspan also often uses Call User Data of 01.

For all other host types, the Gateway supplies no Call User Data by default. Call User Data typically associated with Galileo is: fe 12 11 00 91 00 00 00 00 00 00 00 00 01 00 00. Call User Data typically associated with Apollo is: d3 54 57 4e.

NO_INITIAL_CALL applies to SVC channels only. This option prevents the ALC Gateway from placing an initial call to establish an SVC connection. If this option is specified, the initial call is not placed until a user sends data to the airline host or the airline host has CRT or printer data to send to the user.

The ACTIVITY_TIMER applies to SVC channels only. If set to a non-zero value, this option causes the ALC Gateway to clear the SVC connection after the specified number minutes of continuous inactivity. The default is 10 minutes. If ACTIVITY_TIMER is set to 0, the Gateway will never clear the SVC connection due to inactivity.

Notes on Configuring Multiple IA’s on a single X.25 Connection:

Note 1: When configuring the gateway for X.25, the number of virtual circuits that are set up is the same as the number of configuration files that are specified (with the '-f' option) when starting the ALC gateway. For example:

```
> iate_server -vff -fscfg1.x25 -fscfg2.x25 -fscfg3.x25
```

would direct the gateway to set up three virtual circuits. (This does not change the way a client establishes a connection to a TA.)

Note 2: Non-Sabre PAD_TYPEs that require multiple DATA_IAs need to specify multiple configuration files, one configuration file for each DATA_IA. Each configuration file specifies one DATA_IA and all its associated TA's and other parameters.

Note 3: The Sabre PAD_TYPE can support multiple DATA_IAs per virtual circuit. On Sabre, a maximum of 60 TAs may be configured on one virtual circuit.

X25 Gateway Configuration

The X.25 configuration file is an ASCII text file containing the following five types of lines. The options described here can be specified in any order. Sample Gateway configuration files, such as "x25.cfg1", are provided in the iate_server installation directory.

1. **A comment line.**

   This is any line beginning with the number sign (#). The X.25 Gateway will ignore any comment lines in the configuration file.

   ```
   # This is a test X.25 configuration for SVCs & PVCs on the same line
   ```

2. **A switch:** a line that controls a two-state X.25 configuration option that is either "On" (enabled) or "Off" (disabled).

   These "switch" lines turn on or off X.25 configuration options. Each switch can be specified with an argument value of '+' or '-', or omitted to use the default setting.

   NOTE: For the X.25 Gateway, the '+' or '-' symbol must precede the option name in each switch line.

   For example:

   ```
   + CTS
   ```

   The available switch options are:

Switch	Meaning, if '+' *Meaning, if '-' (shown in italics)*	Default
CTS	CTS must be present for X.25 line to be active *CTS is ignored*	-
DCD	DCD must be present for X.25 line to be active *DCD is ignored*	+
DSR	DSR must be present for X.25 line to be active *DSR is ignored*	-
FLAG	LAPB/HDLC flags required for line to be active *Flags are ignored*	-
INTERNAL_CLOCK	Internal clocking (baud rate must be supplied) *Modem or external device supplies clocking*	-
START_SABM	Line starts by sending SABM	+

	Line starts by sending DM	
EXTENDED_LAPB	Gateway uses LAPB extended addressing *Gateway uses normal LAPB addressing*	-
DCE	Gateway’s LAPB address is logical DCE *Gateway is addressed as a logical DTE*	-
DTE	Gateway’s LAPB address is logical DTE *Gateway is addressed as a logical DCE*	+
	Note: Specify either DCE or DTE, not both. For example, to select DCE operation, specify either “+ DCE” or “- DTE”	
EXTENDED_PACKET	Gateway uses packet-level extended addressing *Gateway uses normal packet addressing*	-
PACKET_SIZE_NEGOTIATION	Gateway permits packet size negotiation *Gateway rejects packet size negotiation*	+
WINDOW_SIZE_NEGOTIATION	Gateway permits packet-window size negotiation *Gateway rejects packet-window size negotiation*	-

3. **Special identifier.**

   This special identifier does not have to be specified. The X.25 Gateway treats it as a comment. It is used to mark the point in the file beyond which the LAPB-related options are grouped together.

   Identifier

   ```
   *LAPB
   ```

4. **A single-value configuration item:**

Item	Parameter	Default	Meaning
LINE_SPEED	1200, 2400 4800, 7200, 9600, 19200, 38400, 56000, or 64000	0	Sets the baud rate for internal clocking. This parameter is required only if INTERNAL_CLOCK is specified. (The default value of 0 is appropriate only for external clocking, and need not be specified in the configuration file.)
N2	*value*	2	Sets the LAPB counter N2.
K	*value*	7	Sets the LAPB value K.
T1	*value*	10	Sets the LAPB timer T1.
T2	*value*	1500	Sets the LAPB timer T2. (*Note:* This timer is specified in milliseconds. All other timers listed here are specified in seconds.)

`T3`	*value*	10	Sets the LAPB timer T3.
`T4`	*value*	10	Sets the LAPB timer T4
`T10`	*value*	60	Sets the packet-level timer T10.
`T11`	*value*	120	Sets the packet-level timer T11.
`T12`	*value*	60	Sets the packet-level timer T12.
`T13`	*value*	60	Sets the packet-level timer T13.
`T24`	*value*	0	Sets the packet-level timer T24.
`PACKET_SIZE`	*value*	256	Sets the X.25 packet size. (*Note:* Set `PACKET_SIZE` only as large as is needed. Available buffer space varies inversely with this parameter.)
`WINDOW_SIZE`	*value*	2	Sets the X.25 packet window size.
`SERVICE`	*string*	`x25gate`	The name of the service over which the X.25 Gateway and the ALC gateway communicate (this must match a service name in the **/etc/services** file).

5. **A range (two-value) configuration item.**

Item	Parameters	Defaults	Meaning
`PVC`	*low high*	0 0	Sets the LCN (Logical Channel Number) range for PVCs, **in decimal, not hex**.
`SVC`	*low high*	1 3	Sets the LCN (Logical Channel Number) range for SVCs, **in decimal, not hex**.

Logical channel numbers (LCNs) range from 1 to 4096. The default range "0 0" for PVCs means that *no* PVCs are available by default. Contact the airline host or X.25 network administrator to determine the correct ranges required for the PVC and/or SVC channels that will be used. [When the DCE (usually the host) originates a call, it calls on the lowest available LCN. When the DTE (usually the gateway) originates a call, it calls on the highest available LCN.]

For SABRE connections using SVCs, the SVC range **must** include at least **TWO** SVCs.

Printer Configuration

A printer may be attached to a serial port on the SPARCstation. (Note: The SPARCstation 10 and Classic each have a single RS-232 DB-25 connector, which can be split into two RS-232 ports through a special Sun cable. With the InnoSys adaptor cable described in Appendix B, the connector supports a single port, **/dev/ttya**.)

The IATE printer program (**iate_printer**) supports three types of print jobs: host prints, screen prints, and session prints. Host prints consist of traffic from a host to a TA which is configured at the host and at the ALC Gateway as a printer. A screen print is an image of a terminal screen. The printer program arbitrates between host and screen prints, tracks the state of the printer, sends text to the printer, etc. Host prints do not go through the UNIX print spooler.

The IATE terminal software may also be used to generate session prints. The terminal software can capture traffic between the host and the terminal to a file, and the resulting session transcript file can be printed through the UNIX print spooler.

Before starting printer installation and configuration, determine which existing port device will be used to attach the printer — for example, **/dev/ttyb**. There should be no login or port administration process associated with a port used for host and/or screen prints. Under Solaris 2.x, check **/etc/inittab** to make sure that it does not contain a line invoking **ttymon** on that port. Under Sun O.S. 4.x, check for a line in the **/etc/ttytab** file referring to the port, and make sure that the rightmost field in that line is set to “off” — for example:

```
   ttyb "/usr/etc/getty std.9600" off
```

See Appendix B for information on printer cabling.

Each Host/Screen printer requires a separate running copy of the printer program (**iate_printer**). Each instance of the printer program is controlled by a Printer Configuration File. The file name must be specified on the command line, as a parameter to the required option -f. A Printer Configuration File is a normal text file editable with any ASCII text editor (such as **vi** or **ex**). Unlike the other configuration files described earlier in this manual, it is not broken down into titled sections — it consists of option specifications only . Two sample printer configuration files are provided: **cfgxonxoff** is a typical configuration file for an XON/XOFF (start/stop) printer; and **pcfg** is a typical configuration file for an RTS/CTS (ready/busy) printer.

Below is a list of the options that may be specified in a Printer Configuration file. Some, but not all of these options include a parameter value.

NAME and USE are required. A printer designated as SCREEN can only be used for screen prints from the IATE terminal emulator. A printer designated as HOST can only be used for host prints. A printer designated as SHARED may be used for both host and screen prints. In the case of a SHARED printer, the **iate_printer** program manages arbitration between host and screen prints.

If no other options are provided, the port defaults to **/dev/ttyb** and the port settings will remain unchanged: the port will operate under whatever serial I/O parameters were configured on that port *before* the current instance of the printer program started.

Option	Parameter	Description
OBJECT_NAME	*Name*	The object name of this printer, it must exactly match a name in the ALC Gateway configuration file
USE	SCREEN or HOST or SHARED	The print job type for this printer. (SHARED specifies that the printer can be used for both host and screen prints.)
PORT_NAME	*port name*	Name of port to which printer is attached (such as **/dev/ttyb**).
FILE_NAME	*file spec*	Specify stdout to pipe the output to another application or specify a valid file name to write the output to a file.
FINAL_EOMS	EOMC, EOMI, EOMU, or EOMPB	Specify one or more End of Message character(s) which cause the printer program to consider a message complete (finished).
BAUD	110, 300, 1200, 2400, 4800, 7200, 9600, 19200, or 38400	Baud rate for printer connection.
STOP_BITS	1 or 2	Number of stop bits per character.
CHARACTER_SIZE	5, 6, 7, or 8	Number of data bits per character.
PARITY	ODD or EVEN or NONE	Character parity option.
ASSUME_ONLINE		Tells the printer driver to send data to the printer regardless of the state of the printer.
RTS_CTS		Use RTS/CTS (“ready/busy”) flow control the printer.
XON_XOFF		Use XON/XOFF (“start/stop”) flow control. (*Note:* Specify either RTS_CTS or XON_XOFF, not both. Results are not defined if both are specified.)
DISABLE_FLOWCONROL		Turns off flow control monitoring.
DEFAULT		Selects BAUD 4800, PARITY ODD, CHARACTER_SIZE 7, STOP_BITS 1, RTS_CTS.
CR_TO_CRLF		“Automatic linefeeds”: Add a Line Feed character after each

		Carriage Return character in print messages.
CRLF_TO_CR		Remove the line feed character from the data stream when a carriage return/line feed sequence is received.
CR_TO_LF		Convert each carriage return character (0x0d) received to the line feed character (0x0a).
SPACE_BEFORE_CR		Add a Space character preceding each Carriage Return character in print messages.
SPOOL_HOST		Send host prints to the spooler.
SPOOL_LOCAL		Send local (screen) prints to the spooler.
SPOOL_ALL		Send local (screen) prints and host prints to the spooler.
SPOOL_PNAME	*printer name*	If this option is set and SPOOL_CMD is *not* set, the printer program will use its default print command (**lpr** ...) to print on the specified printer. One of SPOOL_LOCAL, SPOOL_HOST, and SPOOL_ALL must also be specified.
SPOOL_CMD	*command name*	User-supplied print spooling command: if specified, this must contain the entire command necessary to print (except for the filename to be printed). If this option is specified, the printer program ignores the SPOOL_PNAME setting if present. One of SPOOL_LOCAL, SPOOL_HOST, and SPOOL_ALL must also be specified.
COLLECT_ALL		Collect an entire message from the host before beginning to print the message.
DONT_SUPPLY_FF		Disable the automatic insertion of Form Feeds. If this option is *not* specified, the printer program inserts Form Feed (FF) characters as needed to begin each message at the top of a printed page.
REPLACE_FFS	*number*	Substitute the specified number of Line Feed characters for each Form Feed character in print messages.

`NO_TAB_EXPANSION`		On receipt of the ALC TAB character (value 2C hexadecimal) in a print message from the host, convert it to ASCII TAB (value 9) for printing. If this option is *not* specified, the printer program converts the ALC TAB character to a series of ASCII Space characters sufficient to reach the next tab stop.
`TABS`	`AIRLINE` or `AGENCY`	For Apollo hosts only: Vertical Tab handling option. If the `AIRLINE` setting is specified, the printer program converts ALC character value 3C to ASCII Form Feed. If `AGENCY` is specified, the program converts 3C to Carriage Return. `AIRLINE` is the default. The `X3C_TO_ETX` option, if specified, will override the `TABS` option.
`X3C_TO_ETX`		For Apollo hosts only: If this option is specified, then on receipt of ALC character value 3C hexadecimal in a print message from the host, the printer program will convert it to ASCII ETX (03). If this option is *not* specified, the program will convert 3C to Form Feed or Carriage Return as required depending on the `TABS` option setting.
`NO_TA_TIMEOUT`		Gateway will not time out the connection due to inactivity.
`NO_HEARTBEAT`		Gateway will not terminate the connection if heartbeats are not received.
`SECS_DELAY`		Number of seconds to delay before identifying a printer as off line.
`ACK_FINAL_AFTER_XFER`		Acknowledge a block that ends with one of the FINAL_EOMS after it has been printed.
`TIMESTAMP`		Prefix each message with a timestamp before printing the message to a disk file.
`SEPERATE_MESSAGES`		When printing to disk, each message is printed to a new file.
`MESSAGE_MARK`		Put a delimiter after each message. The delimiter is specified in hex.

`DISABLE_ACKS`		Tell the gateway not to send an ACK even when it receives a SEND_ACK from the printer program.
`INTERCEPT_ROTTY`		Intercept ROTTYs. This option is for Air New Zealand only.
`IGNORE_NZFLUSH`		Ignore the printer “flush” message from an Air New Zealand host system
`MESSAGE`		Assume printer is used to print messages only (not tickets). This enables special message handling for Air New Zealand.
`TICKET`		Assume printer is used to print tickets only. This option is for Air New Zealand only.
`PRINTER_TYPE`	*printer-spec*	Describes the type of ATB2 printer being used. For SABRE, use “SABRE_ATB2”.
`AUTO_ANSWER`	`ON` *or* `OFF`	Allows/prevents the gateway from acknowledging segments on behalf of a client application.
`HEADER_LENGTH`	*number of characters*	The number of characters the printer program should strip off the front of messages coming from the host and then put back onto messages being sent to the host. The default is 0.

The following options are required in the configuration file for a SABRE ATB2 printer:

```
PRINTER_TYPE        SABRE_ATB2
ASSUME_ONLINE
AUTO_ANSWER         OFF
DISABLE_FLOWCONTROL
```

Multiple host printers may be defined by providing a separate configuration file for each printer. A separate run of the printer program is done for each configuration file. For example, the following command will start the printer program using the sample configuration file **pcfg** supplied on the distribution tape. (The ampersand is included so that the program will execute in the background.)

```
iate_printer -fpcfg &
```

Starting a Gateway or Printer

Once installation and configuration have been completed, a gateway or print handler program can be started. This section describes numerous ways to start a gateway or print handler program. (For information about starting the IATE terminal, please refer to the IATE terminal user's manual.)

Starting and Stopping the X.25 Gateway

If an X.25 connection is being used, the X.25 Gateway must be started before the ALC Gateway. The Gateways may be started by anyone; super-user privileges are not required in order to start them.

The X.25 Gateway can be started by simply invoking a shell script. The usage of shell scripts is described later in this document. The Gateway can also be started “manually”. Manual startup is described first because it is a good idea to become familiar with the X.25 Gateway's command-line options.

To start the X.25 Gateway manually, first change to the **x25_gate** subdirectory of the IATE product directory. For example, if the IATE product directory is **\home\iate_product**, then enter:

```
cd \home\iate_product\x25_gate
```

Once there, enter this command to start the X.25 Gateway:

```
x25gate
```

The X.25 Gateway accepts the following command line options:

Option	Description	Default value
`-f`*name*	Specify the name of the gateway configuration file. (Required.)	No default; this option must be specified.
`-b`*name*	INSCC board device file name. It must be specified in the format “/dev/innosbx” where “x” is the board number.	`/dev/innosb6`
`-l`*name*	INSCC X.25 onboard software file name.	`x25.bin`
`-v`*xx*	Set debugging message level *xx*	0 (minimal debugging
output)		
	(in hexadecimal, from 0 to `ff`). (*Note:* Debugging output is not available if the “`d`” option is used.)	
`-d`	Detach user's TTY. Use with “&” at	

end of command line to execute gateway in background. (When running detached in the background, the gateway may be called a *“daemon”*.)

`-h` Display the list of options.

For example, suppose that the X.25 Gateway resides in the **/home/iate_product/x25_gate** directory and the Gateway will be started on board 0 with a configuration file named **x25.cfg** that exists in that same directory. Also, the Gateway is to output all available debugging and diagnostic messages. The following commands could be entered to start the Gateway:

```
cd /home/iate_product/x25_gate
./x25gate -vffff -f./x25.cfg -b/dev/innosb0
```

(The **./** notation is necessary only if the current command search path does not include the current working directory.)

If the Gateway is to be run as a “daemon” — in the background, without displaying any messages — then a proper entry would be:

```
cd /home/iate_product/x25_gate
./x25gate -d -f./x25.cfg &
```

If an X.25 Gateway is running in the background, it can be stopped it by invoking (as super-user) the **stopserver** script described below.

If an X.25 Gateway is running in the foreground, it can be stopped in either of two ways: (**1**) the **stopx25gate** script described below can be invoked (as super-user) — from a separate shell window on the workstation display, for example. Or, (**2**) `Ctrl-C` can be pressed in the window in which the Gateway is running. After pressing `Ctrl-C`, answer “y” to the query as to whether the Gateway should really be stopped.

Loading the ALC Line Driver (INSCC-S Onboard Software)

The following procedure must be followed **before** manually starting the ALC Gateway for direct connection to an ALC line. This procedure loads ALC interface support software into the INSCC-S board. (There is no corresponding procedure required for X.25, because the X.25 Gateway performs this loading function automatically.)

1) Change to the **alc_line** subdirectory under the main IATE product installation directory. For example, if the IATE product files have been installed into the directory **/home/iate_product**, enter:

```
cd /home/iate_product/alc_line
```

2) Use the **loadalc** command to load the ALC line driver software into the INSCC-S board RAM:

```
loadalc
```

Alternatively, a shell script can be run that will start both the line driver and the Gateway.

This script is described later in this document. In this script is run, the foregoing procedure is not necessary. Depending on how the gateway machine is set up, it may be necessary to be super-user to run the loadalc script.

Starting and Stopping the ALC Gateway

The ALC Gateway may be started by anyone; super-user privileges are not required in order to start it.

The ALC Gateway can be started by simply invoking a shell script. The usage of shell scripts is described later in this document. The Gateway can also be started “manually”. Manual startup is described first because it is a good idea to become familiar with the ALC Gateway’s command-line options.

Before manually starting the ALC Gateway for a direct connection to an ALC line, it is necessary to start the ALC line driver as described above. **IMPORTANT:** The ALC Gateway does not automatically load the ALC line driver. (This is different than the X.25 Gateway, which does load its line driver automatically.) In particular, when switching a gateway card from X.25 to ALC, it is very important to remember to load the ALC line driver - otherwise the gateway will probably CRASH.

After starting the ALC line driver, change to the **alc_gate** subdirectory of the IATE product directory. For example, if the IATE product directory is **\home\iate_product**, enter:

```
cd \home\iate_product\alc_gate
```

Once there, enter this command to start the ALC Gateway:

```
iate_server
```

The ALC Gateway accepts the following command line options:

Option	Description	Default value
-f*name*	Specify the name of the gateway configuration file. (Required.) The config file should be in the same directory as the gateway software.	No default; this option must be specified.
-F*name*	Specify the name of a file containing a list of configuration file names.	no default
`-l`*name*	Set Gateway debug output level to *xxxx* (see `-v` below) and write the output to a file named “server.log”. Both `-l` and `-v` may be be specified as long as neither is set to ffff. Another way to save the log to a file is to “redirect” it. For example: iate_server -vff -fscfg > logfile1	no default
`-v`*xxxx*	Set Gateway console debug output level to *xxxx* (in hexadecimal, from 0 to `ffff`). Set specific bits to show the following types types of debugging information:	0 (minimal debugging output)

	0x0001 Network data transfer activities. 0x0002 Gateway initialization phase. 0x0004 Gateway-specific activity. 0x0008 Client-specific activity. 0x0010 Activity specific to X.25. 0x0020 Text of data messages. 0x0040 Errors (severe). 0x0080 Warnings (less severe). 0x0100 Timer-related activity. 0x0200 Network activity specific to the TLI transport interface. 0x1000 Buffer management activity. 0x2000 Buffer management debugging. 0x8000 Very verbose output. (*Note:* Debugging output is not available if the "d" option is used.)	
-d	Detach user's TTY. Use with "&" at the end of command line to execute the gateway in the background. (When running detached in the background, the gateway may be called a *"daemon"*.)	
-p	Sets the number of seconds the gateway pauses before it starts up.	0
-h	Display the list of options.	

For example, suppose that the ALC Gateway resides in the **/home/iate_product/alc_gate** directory, and the Gateway will be started with a configuration file named **alc.cfg** that exists in that same directory. Also, the Gateway is to to output all available debugging and diagnostic messages. The following commands could be entered to start the Gateway:

```
cd /home/iate_product/alc_gate
./iate_server -vffff -f./alc.cfg
```

(The **./** notation is necessary only if the current command search path does not include the current working directory.)

To start the ALC gateway with two configuration files (for example, two ALC connections, or an ALC connection and an X.25 connection, or one X.25 connection with multiple virtual circuits) :

```
iate_server -vff -fsabre.cfg -famadeus.cfg
```

If the Gateway is to be run as a "daemon" — in the background, without displaying any messages — then a proper entry would be:

```
cd /home/iate_product/alc_gate
./iate_server -d -f./alc.cfg &
```

If an ALC Gateway is running in the background, it can be stopped it by invoking (as super-user) the **stopserver** script described below.

If an ALC Gateway is running in the foreground, it can be stopped it in either of two ways: (**1**) the **stopserver** script described below can be invoked (as super-user) — from a separate shell window on the workstation display, for example. Or, (**2**) `Ctrl-C` can be pressed in the window in which the Gateway is running. After pressing `Ctrl-C`, answer “y” to the query as to whether the Gateway should really be stopped.

Using Shell Scripts to Start and Stop the Gateways

A script named **startix25** is provided to start both the X.25 and ALC Gateways in the background. It may be interesting to examine this script. **startix25** first invokes two other scripts, **stopserver** and **stopx25gate**, to ensure that any previous invocations of the Gateways are no longer running. Then, **startix25** starts the X.25 Gateway, and finally the ALC Gateway using the configuration file **scfg.x25**. (Note that the X.25 Gateway will load X.25 interface software into the INSCC-S board, so an explicit board loading command is not needed in the script file.)

Another supplied script, named **startialc**, starts the ALC Gateway for a direct ALC connection (not X.25). This script starts by invoking **loadalc** to load the ALC interface software into the INSCC-S board. This step is necessary because, unlike the X.25 Gateway, the ALC Gateway does not automatically load the board.

Another supplied script, named **stopx25gate**, kills all **x25gate** processes. Similarly, **stopserver** kills all **iate_server** (ALC Gateway) processes. **stopx25gate** and **stopserver** should only be executed by the super-user (to ensure sufficient user privileges to kill all Gateway processes system-wide).

It may be necessary to create specialized scripts. For example, it may be desirable to create a script to start the X.25 Gateway only, or a script to invoke one or both Gateways using customized configuration files.

Starting and Stopping the Print Handler

A script named **startprinter** is provided to start a print handler in the background using the printer configuration file **pcfg**. The print handler may be started by anyone; super-user privileges are not required in order to start it.

iate_printer can also be started by hand. The command line options are:

Option	Description	Default value
`-f`*name*	Specify the name of the printer configuration file. (Required.)	No default; this option must be specified.
`-v`*xx* output)	Set debugging message level *xx* (in hexadecimal, from 0 to `ff`). (*Note:* Debugging output is not available if the “`d`” option is used.)	0 (minimal debugging
`-d`	Detach user’s TTY. Use with “&” at end of command line to execute program in background as a “daemon”.	

`-h`	Display the list of options.

For example, the supplied **startprinter** script uses the following command to start **iate_printer**, which is in the current working directory at the time this command is executed. **iate_printer** will use the printer configuration file **pcfg**. The program will run in the background as a “daemon” and will not print messages to the user’s TTY.

```
./iate_printer -f./pcfg -d &
```

Each invocation of the print handler handles printing for one IA and TA only. To permit printing to multiple printer TAs, start multiple copies of **iate_printer** using different printer configuration files (in place of **./pcfg** in the example above).

If the **iate_printer** program is to be started by anyone other than the super-user, it will be necessary to change the permissions on the TTY device driver file. For example, if **iate_printer** is configured to use **/dev/ttyb**, the super-user must execute a command such as:

```
chmod 777 /dev/ttyb
```

The supplied script named **stopprinters** kills all **iate_printer** processes on the machine where **stopprinters** is invoked. **stopprinters** can only be executed by the super-user. The super-user can also use the following commands to find and stop an individual **iate_printer** process under Solaris. For Sun O.S. 4.x, use `ps -a` instead of `ps -e`.

```
ps -e | grep iate_printer
    (... output shows any current iate_printer process numbers ...)
kill -9 process_number
```

Quick Start-Up (for an ALC-only configuration)

The following commands can be used to start the ALC Gateway and Host Printer Application after the machine has been rebooted or configuration has been changed. This example assumes that the “innosb” device drivers have already been installed and loaded, and that the IATE package has been installed into the directory **/home/iate_product**.

```
# cd /home/iate_product
# startialc
# startprinter
```

Supplemental Programs

The following diagnostic and monitoring programs are provided with the ALC Gateway package. They may be run by any user on any machine with network access to the machine running the ALC gateway.

serverwatch

The **serverwatch** utility issues an ongoing report of Gateway activity. The report contains "debugging" messages retrieved from the Gateway. By default, **serverwatch** displays its output on the TTY display. **serverwatch** can also log activity information to 2 different files, switching from one file to the other after every 2000 lines of output. To stop the **serverwatch** utility, press `Ctrl-C`.

The command line options are:

Option	Description	Default value
`-c0` *or* `c1`	Select whether or not to show activity information on the TTY display.	`c1` (to display the information)
`-l`*name*	Log to files with prefix *name*.	
`-v`*xxxx*	Gateway debug output level *xxxx* (in hexadecimal, from 0 to `ffff`). Set specific bits to show the following types of debugging information: `0x0001` Network data transfer activities. `0x0002` Gateway initialization phase. `0x0004` Gateway-specific activity. `0x0008` Client-specific activity. `0x0010` Activity specific to X.25. `0x0020` Text of data messages. `0x0040` Errors (severe). `0x0080` Warnings (less severe). `0x0100` Timer-related activity. `0x0200` Network activity specific to the TLI transport interface. `0x1000` Buffer management activity. `0x2000` Buffer management debugging. `0x8000` Very verbose output.	`0x000D` (the `0x0001`, `0x0004`, and `0x0008` bits)
`-h`	Display the list of options.	

testterm

testterm is a VERY simple terminal emulator that can be used to test the connection between the gateway and the airline host. **testterm** displays a System Available/Unavailable indication and a keyboard Locked/Unlocked indication. A command can be sent to the host by typing the command in from the keyboard and pressing the Enter key. To unlock the keyboard, press the exclamation point (“!”) key.

The command line options are:

Option	Description
-o@*hostname*\\servname\\object	Specify the ALC Gateway by its host machine’s name, the TCP port service name from **services**, and the object name. Note the required double-backslashes. @*hostname*\\ is only required when the ALC gateway is running on a different machine than **testterm**. *servname*\\ is only required when the default service, 1413, is not being used. *object* is always required.
-h	Display the list of options.

Example:

```
testterm -oterm1a
```

showcfg

The **showcfg** utility displays current configuration information from an ALC Gateway. If the command `showcfg` is entered alone (without any command-line parameters), the program will look for a Gateway on the local machine using the default service name **ialcserver**, and will display its configuration. To specify a different Gateway on the local machine, or a Gateway that is running on a remote machine, use the `-g` option. For example, suppose a Gateway is running on a remote machine named **gatesys**, using a TCP port which has been assigned service name **ialcserver** in the **/etc/services** file on the local machine. In that case, the following command will display configuration information for that remote gateway:

```
        showcfg -g@gatesys\\
```

The `@` symbol before the host name is required, as are the trailing backslashes. No spaces are allowed in the option string that begins with `-g`.

To access a Gateway using a different service name (other than **ialcserver**), both the gateway name *and* the service name must be specified. For example, suppose a second Gateway on the **gatesys** machine uses a TCP port which has been assigned service name **ialcsserver2**. The following command will display configuration information for that gateway:

```
        showcfg -g@gatesys\\ialcserver2\\
```

The command line options are:

Option	Description
-g@*hostname*\\servname\\	Specify the ALC Gateway by its host machine’s name

and the TCP port service name from **/etc/services**. Note the required double-backslashes.

If just -g@*hostname*`\\`is specified, the program will look for a Gateway with service name `ialcserver` on the specified host machine. If this option is not specified at all, the program will look for a Gateway with service name `ialcserver` on the local machine.

`-c` Display connected objects only.

`-h` Display the list of options.

showdef

The **showdef** program displays the defaults for a given host type. To use it, enter the command:

```
showdef host_type
```

The *host_type* argument can be any one of these: `SABRE`, `PARS`, `APOLLO`, `DATAS`, `SODA`, `SHARES`, or `KLM`. Case is not significant.

showxlat

The **showxlat** program displays translation tables. To use it, enter the command:

```
showxlat host_type order
```

The *host_type* argument can be `sabre`, `pars`, `klm`, `apollo`, or `all`. The host type must be specified. The *order* argument specifies the character code set to order the output; this can be `ASCII`, `alc`, or `line`. If the order is not specified, the program will default to `ASCII` order. Note that lower case is required for both arguments.

showval

The **showval** program displays message names and their numeric codes. Depending on the command line options, the program displays a subset or a complete list of messages that can pass between the ALC Gateway and a client API library, between the ALC Gateway and the printer program, between the ALC Gateway and the ALC onboard software, or between the ALC and X.25 Gateways. **showval** is intended mainly for internal use at InnoSys, but it may sometimes be useful to users who need to understand a numeric error or diagnostic message.

The command line options are:

Option — Description

`0x`*xx* — Specify a single message code in hexadecimal.

nn — Specify a single message code in decimal.

> (Only one numeric option can be specified, in either hexadecimal or decimal. A numeric option must be the first option on the command line. If a numeric option is not specified, the program will display a complete list of messages under the category that was specified by any one of the following options.)

`cs` — Display client/server messages that pass between the ALC Gateway and the IATE API library.

`api` — Display IATE API messages related to the IateControl API function.

`link` — Display IATE API messages related to the IateOpen API function.

`peer` — Display peer-to-peer messages related to the printer program.

`err` — Display error messages that the API library can return to an IATE terminal or API user application.

`alc` — Display messages that pass between the ALC Gateway and the ALC onboard software.

`x25` — Display messages that pass between the ALC and X.25 Gateways.

`all` — Display all types of messages available.

Examples:

To display all possible messages that can pass between an ALC Gateway and an IATE terminal or API user application:

Enter: `showval cs`
Output: *list of message codes and names*

To display the name of the message 0x0035 hexadecimal between ALC & X.25 Gateways:

Enter: `showval 0x0035 x25`
Output: `0x35:CLEARREQUEST`

To display the name of the IATE API error code -2217 decimal:

Enter:	`showval -2217 err`
Output:	`-2217:TooMuchDataQueued`

keyhelp

The **keyhelp** program displays a keyboard map.

There are two keyboard maps supplied with the X IATE product, one for SABRE style keyboards and one for Apollo style keyboards. The keyboard map file for SABRE is named **iatekey.sabre**. The keyboard map file for Apollo is named **iatekey.apollo**. Correspondingly, there are two keyboard help files, **iatehelp.sabre.sun** and **iatehelp.apollo.sun**.
If the **.iatehelp** file has been created at installation (see Distributing User Files), **keyhelp** will display the keyboard help file in **.iatehelp**. To specify a keyboard map explicitly, enter one of the following commands:

```
keyhelp -help iatehelp.sabre.sun
```

or

```
keyhelp -help iatehelp.apollo.sun
```

A space is required after `-help`.

There is one command line option (as was illustrated above):

Option	Description
`-help` *filename*	Specify name of keyboard map help file. **.iatehelp** is the default if this option is not specified.

innostop

This utility disconnects a client from the ALC Gateway. Before running innostop, use `showcfg -c` to find the object number of the client that is to be disconnected.

Option	Description
`-g@`*hostname*`\\`*servname*`\\`	Specify the ALC Gateway by its host machine's name and the TCP port service name from **/etc/services**. Note the required double-backslashes. If only -g@*hostname*\\is specified, the program will look for a Gateway with service name `ialcserver` on the specified host machine. If this option is not specified at all, the program will look for a Gateway with the service name `ialcserver` on the local machine.

`-o`*number* Specify the object-number (in decimal) of the object to disconnect from the ALC Gateway.

-h Display the list of options.

Example:

```
innostop -g@host\\ialcserver\\ -o7
```

innoping

This utility sends repeated requests for an ALC Gateway to respond, and reports each response, at a rate of about 1 request/response cycle per second. This utility is useful in confirming Gateway accessibility: consistent receipt of responses confirms that the Gateway is running, and that it is accessible through the network from the machine on which **innoping** is running. In addition, **innoping** will initially retrieve and display current configuration information from the running Gateway.

(*Note:* Some users may be familiar with **innoping**'s namesake, the UNIX utility **ping**e, the UNIX utility **ping**, whose basic function is somewhat similar: to obtain responses from a remote computer system and confirm that system's accessibility over the network.)

Option Description

`-g@`*hostname*`\\`*servname*`\\` Specify the ALC Gateway by its host machine's name and the TCP port service name from **/etc/services**. Note the required double-backslashes.

If only -g@*hostname*\\is specified, the program will look for a Gateway with service name `ialcserver` on the specified host machine. If this option is not specified at all, the program will look for a Gateway with the service name `ialcserver` on the local machine.

`-n`*number* Repeat the request/response cycle *number* times, with the exception that `-n1` instructs **innoping** to merely report whether or not the Gateway is available, and then exit. If the `-n`*number* option is not specified, the program will continue requesting responses until it is stopped (e.g., press `Ctrl-C`).

`-h` Display the list of options.

Example:

```
innoping -g@host\\ialcserver\\ -n10
```

Board Test

The board test program, insccstest, tests all of the INSCC-S boards it finds in the test machine. It then summarizes the test results on the screen and writes a complete record of the test results to the file named “insccslog*”. There is a loopback adapter included with the gateway board. This adapter should be plugged into the RS-232 port on the INSCC-S board when the Board Test software is run. The software will run without the loopback adapter installed, but it will report errors on all RS-232-related tests.

To run the Board Test software, log on as root, change to the “testboard” directory and enter "insccstest".

Example:

```
insccstest
```

Appendix A — Files

The following list summarizes all of the Gateway software and related files, and additional files associated with other components of the IATE product suite. Not all of the subdirectories listed here may be included in a particular distribution package, depending on which products were ordered. However, within each subdirectory that is included with the distribution package received, all of the files listed for that subdirectory should be present.

Each executable binary program file or shell script file is marked with an asterisk '*'. (The asterisk is not part of the file name.) Other files contain text or binary data, API software libraries, etc.

Directories	Files	Descriptions
iate_product/		Top-level product distribution directory — contains various subdirectories and the following startup/shutdown scripts:
	startialc*	Start ALC line drivers and ALC Gateway
	startserver*	Start the ALC Gateway in the background
	stopserver*	Stop the ALC Gateway
	startprinter*	Start the print handler in the background
	stopprinters*	Stop all instances of the print handler
	startix25*	Start the X.25 and ALC Gateways
	stopx25gate*	Stop the X.25 Gateway
iate_product/alc_gate/		The ALC Gateway
	iate_server*	The ALC Gateway program
	scfg	Gateway configuration file
	scfg.sabre	Sample configuration file for SABRE
	scfg.apollo	Sample configuration file for Apollo
	scfg.x25.sabre	Sample configuration file for SABRE using an X.25 connection *(Additional sample configuration files may be included with the distribution)*
iate_product/alc_line/		ALC "onboard" interface software for INSCC-S boards
	loadalc	Shell script to load line driver to INSCC-S board RAM
	loadboard	Loader program (invoked by the script above)
	alcmulti.com	ALC Line Driver: INSCC-S onboard software

iate_product/x25_gate/	The X.25 Gateway and “onboard” software
x25gate	The X.25 Gateway program
x25.bin	X.25 Line Driver: INSCC-S onboard software (loaded automatically by the X.25 Gateway)
x25cfg1	An X.25 Gateway configuration file that will use network service x25gate1
x25cfg2	An X.25 Gateway configuration file that will use network service x25gate2
iate_product/printer/	The Host Print and Local Print handler
iate_printer*	The print handler program
pcfgrts	Sample configuration file for RTS/CTS (“ready/busy”) printers
pcfgxon	Sample configuration file for XON/XOFF (“start/stop”) printers
options	A list of configuration options for the printer program
iate_product/scripts/	INSCC-S (offboard) device driver
readme	Script to load device driver
iate_product/device/	INSCC-S (offboard) device driver
loaddev*	Script to load device driver
innosb	Loadable device driver file
show_innosb*	A utility to display the available device names on the system, i.e. /dev/innosb0
iate_product/utils/	Utility programs
serverwatch*	Show ALC Gateway activity
showcfg*	Show ALC Gateway configuration
showdef*	Show defaults for a given host type
showval*	Interpret message and error codes
showxlat*	Show translation tables
innoping*	Check ALC Gateway status
innostop*	Forcefully disconnect a client
dumpxlat*	Utility to dump the current translation table as part of the ALC gateway diagnostics
setdebug*	Utility to set the diagnostic debugging level of the ALC gateway
setlog*	Utility to set the gateway logging level
setlogflush*	Utility to set the number of lines after which the gateway flushes to the disk the internal buffer holding log information
setlogline*	Set the maximum number of lines to which a log file can grow.When the maximum is

	exceeded the gateway will start writing at the beginning of the file
`testterm*`	Terminal application used to open a test connection to the ALC gateway

`iate_product/xiate_term/`	The X version of the IATE terminal emulator
`xiate*`	The terminal emulator program
`XIATE`	Terminal initialization data file (*not* executable)
`iatekey.pars`	Pars keyboard map - binary
`iatekey.apollo`	Apollo keyboard map - binary
`iatekey.sabre`	SABRE keyboard map - binary
`keyhelp*`	Utility to display keyboard maps
`iatehelp.apollo.sun`	Keyboard help file for Apollo keyboard
`iatehelp.sabre.sun`	Keyboard help file for SABRE keyboard

`iate_product/keyboard/`	source & tools for keyboard maps
`readme`	information about keyboards
`coviakey`	Apollo keyboard source file (editable text)
`sabrekey`	SABRE keyboard source file (editable text)
`makefile`	used to compile “makekeys.c” to “makekeys”
`makekeys.c`	keyboard map compiler source*
`makekeys`	keyboard map compiler executable* (Solaris)
`generics.h`	file used by makekeys.c
`keys.h`	file used by makekeys.c
	* the keyboard map compiler translates keyboard map source text files into binary keyboard map files for the terminal’s use

`iate_product/curses_term/`	The Curses version of the IATE terminal emulator
`iate*`	The terminal emulator program
`iate.cfg`	Terminal emulator configuration file
`iate.pf`	Programmable Function Keys file
`iate.act`	Special Action Keys file
`htable`	Host Table file
`termkeys`	Keyboard Map file

`iate_product/fonts/`	Font files for X IATE. `.bdf`: X11 BDF font bitmap files `.ff`: font family files `.fb`: X11/NeWS font files
`alc6.bdf`	Non-SABRE 6-point font files

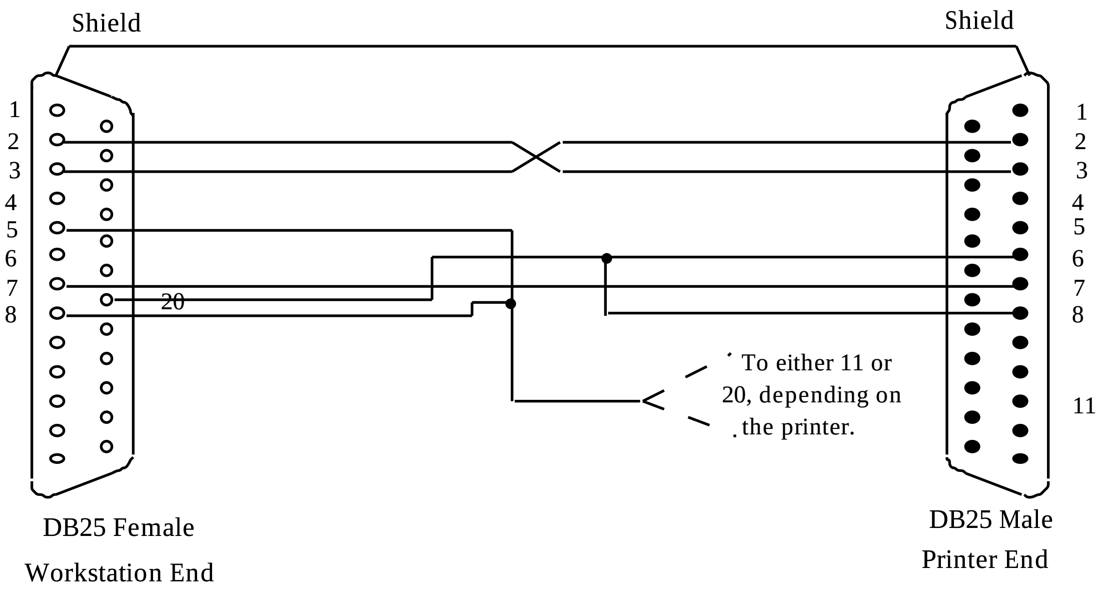
alc6.ff alc610.fb	
alc6B.bdf alc6B.ff alc6B10.fb	Non-SABRE 6-point bold font files
alc8.bdf alc8.ff alc813.fb	Non-SABRE 8-point normal font files
alc8B.bdf alc8B.ff alc8B13.fb	Non-SABRE 8-point bold font files
alc9.bdf alc915.fb	Non-SABRE 9-point normal font files
alc9B.bdf alc9B15.fb	Non-SABRE 9-point bold font files
iate6.bdf iate6.ff iate610.fb	SABRE 6-point normal font files
iate6B.bdf iate6B.ff iate6B10.fb	SABRE 6-point bold font files
iate8.bdf iate8.ff iate813.fb	SABRE 8-point normal font files
iate8B.bdf iate8B.ff iate8B13.fb	SABRE 8-point bold font files
iate9.bdf iate915.fb	SABRE 9-point normal font files
iate9B.bdf iate9B15.fb	SABRE 9-point bold font files
*.pcf.Z.	Solaris-compatible X font files
iate_product/samples/	Sample source code using the IATE API
makefile	Makefile for the following 2 programs
testterm.c	A simple terminal emulator program
sendpeer.c	Program to send peer-to-peer messages
iocalls.c	Source file required to compile sendpeer program
testmulti.c	Source code for supporting multiple connections to the gateway
testmulti*	Program for supporting multiple connections to the gateway
testterm*	Terminal application used to open a test

	connection to the ALC gateway
iate_product/lib/	The IATE API library
iatelib.a	API library file
iate_product/headers/	Header files for programs using the API
U_API.h	General definitions
U_APIerr.h	Error-code definitions
U_APItypes.h	Type definitions
U_APImaxs.h	Value-limit definitions
U_CMNmaxs.h	Additional value-limit definitions
U_APIpros.pro	Function prototypes
iate_product/include/	
U-CMNgatecodes.h	Defines the types of gateways
iate_product/monitors/	
alcmon*	ALC line monitor
tx25mon*	Text-based X.25 line monitor
x25_gui/x25mon*	Graphical user interface X.25 line monitor
x25_gui/start_monitor*	Script to start the GUI X.25 line monitor
x25_gui/readme	Documentation on the GUI X.25 line monitor
iate_product/testboard/	
readme	Documentation on the InnoSys INSCC-S serial communication card board test
tstinscc.com	Download image used by inscctest utility
inscctest*	Utility to test the INSCC-S serial communication card

Appendix B — Printer cabling

The cable used to connect the UNIX workstation to a printer is wired as follows:

Workstation-to-Printer Cable Wiring Diagram



Connection Lists

For printer types other than Worldspan

Work-station	Printer
DB25F	DB25M
Shield	Shield
2	3
3	2
5	11,20
7	7
8	11,20
20	6,8

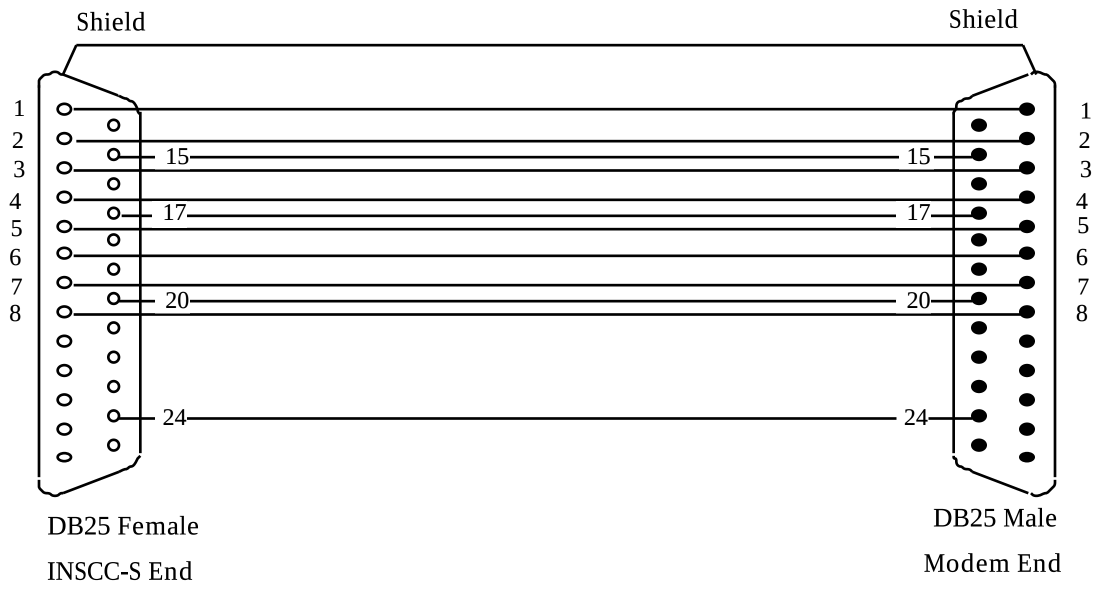
For Worldspan standard printers

Work-station	Printer
DB25F	DB25M
Shield	Shield
2	3
3	2
5	11
7	7
8	20
20	6,8

The cable works with both RTS/CTS (“ready/busy”) and XON/XOFF (“start/stop”) flow control.

Appendix C — Modem cabling

INSCC-S One Port Cable Wiring Diagram



One-Port Cable Connection List

DB25F	DB25M
Shield	Shield
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
15	15
17	17
20	20
24	24

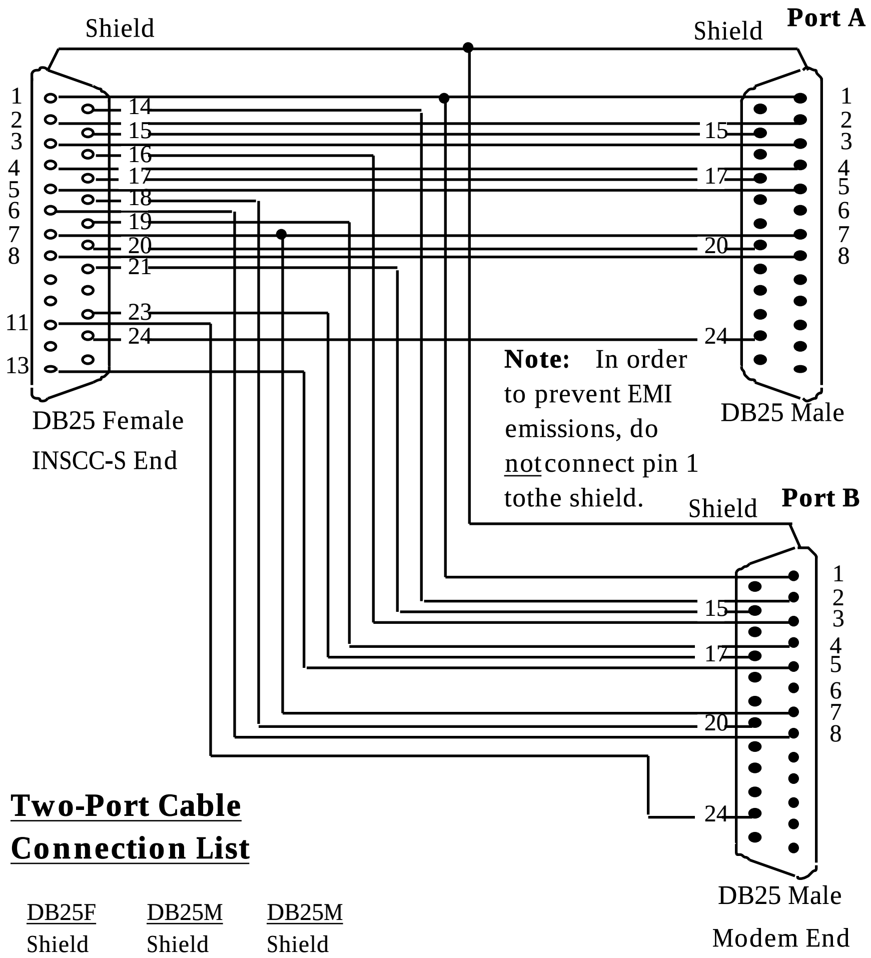
Cable Shielding Specifications

(Required in order to meet stated EMI levels)

- Use cable with braided copper shield.
- Use metal EMI/RFI hoods.
- Make a 360 degree connection between the braided shield and metal hoods .
- Do not connect pin 1 to the shield.

WARNING - A simple “straight thru on all 25 pins” cable may be used with most modems/DSU’s. However, some modems/DSU’s are **very** sensitive to signals on pins other than 1,2,3,4,5,6,7,8,15,17,2 24. Since the INSCC-S card has two ports, signals may be present on pins other than those just listed. such a problem is suspected, **use a custom cable** instead of using a “straight thru on all 25 pins” cable

INSCC-S Two-Port Cable Wiring Diagram



Note: In order to prevent EMI emissions, do not connect pin 1 tothe shield.

Two-Port Cable Connection List

DB25F	DB25M	DB25M
Shield	Shield	Shield
1	1	1
2	2	
3	3	
4	4	
5	5	
6		8
7	7	7
8	8	
11		24
13		5
14		2
15	15	
16		3
17	17	
18		20
19		4
20	20	
21		15
23		17
24	24	

Cable Shielding Specifications

(Required in order to meet stated EMI levels)

- Use cable with braided copper shield.
- Use metal EMI/RFI hoods.
- Make a 360 degree connection between the braided shield and metal hoods .
- Do not connect pin 1 to the shield.

Appendix D — Sample Configuration Files

Several sample configuration files are presented below. These sample configuration files include all the necessary required options for the host specified, but in all cases, the individual values of each configuration item, such as IA, TA, Remote address, etc. need to be changed to correspond to the values configured at the host. In addition, some configurations may require the use of some of the optional entries.

```
#
#       sample Apollo configuration file for the ALC Gateway on an ALC line
#

*HOST TYPE
APOLLO

*HOST CONNECTION
BOARD_NUMBER        0
PORT_NUMBER         0
PORT_NAME                   alc_port_0

*IAS
DATA_IA             1

*OBJECT DEFINITIONS
1       1       TERMINAL    term2101        **
1       11      TERMINAL    term2111        **
1       12      TERMINAL    term2112        **
1       13      TERMINAL    term2113        **
1       14      TERMINAL    term2114        **
1       15      TERMINAL    term2115        **
1       16      TERMINAL    term2116        **
1       19      TERMINAL    term2119        **
1       20      TERMINAL    term2120        **
1       18      PRINTER     prt2118         **
1       02      PRINTER     prt2102         **
```

```
#
#       sample KLM configuration file for the ALC Gateway on an ALC line
#

*HOST TYPE
KLM

*HOST CONNECTION
BOARD_NUMBER        0
PORT_NUMBER         0
PORT_NAME                   alc_port_0

*BROADCAST TAS
1       1c
```

```
*IAS
DATA_IA                                             1

*OBJECT DEFINITIONS
1       1       TERMINAL        term01      **
1       11      TERMINAL        term11      **
1       12      TERMINAL        term12      **
1       13      TERMINAL        term13      **
1       14      TERMINAL        term14      **
1       15      TERMINAL        term15      **
1       16      TERMINAL        term16      **
1       19      TERMINAL         term19             **
1       20      TERMINAL         term20     **
1       18      PRINTER         prt18       **
1       2       PRINTER         prt02       **


#
#       sample SABRE configuration file for the ALC Gateway on an ALC line
#

*HOST TYPE
SABRE

*HOST CONNECTION
BOARD_NUMBER                    0
PORT_NUMBER                     0
PORT_NAME                               alc_port_0

*IAS
DATA_IA                         1

*OBJECT DEFINITIONS
1       2       TERMINAL                term2       **
1       4       TERMINAL                term4       **
1       12      PRINTER                 prt12       **


#
#       sample Apollo configuration file for the ALC Gateway on an X.25 line
#

*HOST TYPE
APOLLO

*HOST CONNECTION
X25_GATEWAY                     x25gate
PAD_TYPE                        APOLLO
CUD                             d3 54 57 4e
REMOTE_ADDRESS                          2121374
SOURCE_ADDRESS                          2101765

*IAS
DATA_IA         1
```

```
*LINE NUMBERS
1 00

*OBJECT DEFINITIONS
1 1  F16D1500         TERMINAL   term1       **
1 2  F16D1600         TERMINAL   term2       **
1 3  F16D1700         TERMINAL   term3       **
1 4  F16D1800         TERMINAL   term4       **

#
#        sample Galileo configuration file for the ALC Gateway on an X.25  line
#

*HOST TYPE
GALILEO

*HOST CONNECTION
X25_GATEWAY              x25gate
PAD_TYPE                 WORLDSPAN
REMOTE_ADDRESS           9001062
LOCAL_ADDRESS            2614735
IA_NATIVE_8_BIT
CUD                      fe 12 11 00 91 00 00 00 00 00 00 00 00 01 00 00

*IAS
DATA_IA                  14

*LINE NUMBERS
14 A2

*OBJECT DEFINITIONS
14 12  TERMINAL      term12        **
14 13  TERMINAL      term13        **
14 14  TERMINAL      term14        **
14 03  PRINTER       prt03         **


#
#        sample SABRE configuration file for the ALC Gateway on an X.25 line
#

*HOST CONNECTION
X25_GATEWAY              x25gate
REMOTE_ADDRESS           9188328410

*HOST TYPE
SABRE

*IAS
DATA_IA      06ba

*OBJECT DEFINITIONS
06ba02  TERMINAL   term2         **
06ba04  TERMINAL   term4         **
06ba06  TERMINAL   term6         **
```

```
06ba08  TERMINAL   term8          **
06ba0a  TERMINAL   terma          **
06ba0c  TERMINAL   prtc           **
06ba0e  TERMINAL   prte           **


#
#       sample Worldspan configuration file for the ALC Gateway on an X.25 line
#

*HOST TYPE
PARS

*HOST CONNECTION
X25_GATEWAY              x25gate
PAD_TYPE                 WORLDSPAN
REMOTE_ADDRESS           90012001400170
LOCAL_ADDRESS            26137850676271
CUD                      fe 12 10 00 a1 00 00 00
NO_INITIAL_CALL

*GATEWAY DEFAULTS
SERVER_NAME              ialcserver

*IAS
DATA_IA                  14

*LINE NUMBERS
14 00

*OBJECT DEFINITIONS
14 12  TERMINAL    term12         **
14 13  TERMINAL    term13         **
14 03  PRINTER     prt03          **

#
#       sample Air New Zealand configuration file for the ALC Gateway on an X.25 line
#

*HOST CONNECTION
VIRTUAL_PORT             0
PAD_TYPE                 AIRNZ
X25_GATEWAY              X25gate
DESTINATION_PVC                2

*HOST TYPE
PARS

*IAS
DATA_IA      5014

*OBJECT DEFINITIONS
10  TERMINAL        term10        **
02  PRINTER         prt02         **
2a  PRINTER         prt2a         **
```

```
#
#       sample configuration file for the X.25 Gateway
#
#       this configuration file specifies DCE operation, internal clocking, both SVCs &
#       PVCs, a non-standard X.25 service name, and alternate LAPB parameters
#

DCE
INTERNAL_CLOCK
LINE_SPEED 9600
SVC 4 6
PVC 1 3
SERVICE x25gate1
T1 5
T2 500
T3 10
T4 10
FLAG
CTS
DCD
DSR

#
#       sample configuration file for an X.25 Gateway with the IATA PVC PAD type
#
#       this configuration file specifies internal clocking and a
#       non-standard X.25 service name
#

SERVICE x25gate0
DCE
INTERNAL_CLOCK
LINE_SPEED 9600
PACKET_SIZE 128
PVC 1 4
FLAG
CTS
DCD
DSR

#
#       sample configuration file for a SABRE X.25 Gateway in Japan
#
#       this configuration file specifies a normal SVC range and a large packet size.
#       In Japan, SABRE uses a window size of 7.  This is different than the default (2).
#

SVC 1  3
PACKET_SIZE 512
WINDOW_SIZE 7
FLAG
CTS
DCD
DSR
```

```
#
#       sample configuration file for a Worldspan X.25 Gateway
#
#       this configuration file specifies a normal SVC range and a small packet size
#

SVC 1024 1047
PACKET_SIZE 128
SERVICE x25gate
```

Appendix E — Typical Gateway startup under ALC

The following text is sample output from the ALC gateway (iate_server) when it is connected to an ALC line and started with the debugging set to -vff. Each line starts with a type code (C=Comment, W=Warning, E=Error); queue counts (nn:nn); and a time stamp (dd:hh:mm:ss).

```
@(#)InnoSys IATE_SERVER Version 2.3b.5 (solaris) as of Jun 10 1997 13:02:23

This Gateway is licensed as follows:

   8 SABRE connections

C (17:-1) 10:13:04:59 debug.c:175          {----} Setting Gateway debugging to 0x00ff
C (17:-1) 10:13:04:59 vport.c:339          {----} vport tables has room for 1024 entries
C (17:-1) 10:13:04:59 socket.c:92          {----} Socket table has room for 1024 entries
C (17:-1) 10:13:04:59 config.c:620         {----} Processing configuration file scfg.
C (17:-1) 10:13:04:59 config.c:3195        {----} Validating configuration file scfg.
E (17:-1) 10:13:04:59 config.c:3222        {----} Unrecognized key <*IGNORE> at line 122 in file <scfg>
C (267:-1) 10:13:04:59 vport.c:127         {----} Assigned vport 0 for use by socket type 1:HOST
C (267:-1) 10:13:04:59 config.c:675        {----} Configuring line for protocol SABRE.
C (270:-1) 10:13:04:59 config.c:686        {----} Configuring ALC line on board:0 port:0 vport:0.  Using device
      /dev/innosb0.
C (270:-1) 10:13:04:59 config.c:1607       {----} Will respond to polls with data for polled IA only.
C (690:-1) 10:13:04:59 config.c:770        {----} Line configured for 1 ias, 60 tas, and 0 polling tas.
C (690:-1) 10:13:04:59 vport.c:627         {----} Added vport 0:<alc_board_0> to alc_list

This Gateway has objects configured for the following host types:

   3 SABRE objects

A total of 3 objects are defined.

C (690:-1) 10:13:04:59 qhigh.c:150         {----} Allocated storage for 690 queues of 52 bytes each. (total
      35880)
C (690:-1) 10:13:04:59 qhigh.c:181         {----} Allocated storage for 131 queue_items of 2316 bytes each.
      (total 1598040)
C (448:131) 10:13:04:59 nettli.c:418       {----} Binding ialcserver at port 1413 on host ultra1, ip address
      INADDR_ANY, to socket 4.
C (448:131) 10:13:04:59 socket.c:275       {----} Initializing socket record:4 type:5:LISTEN_SOCK
C (445:131) 10:13:04:59 socket.c:275       {----} Initializing socket record:5 type:1:HOST_DIRECT
C (445:131) 10:13:04:59 hostio.c:284       {----} vport:0 <alc_board_0> opened:/dev/innosb0 as:5

--- Downloading Configuration to INNSCC-S (board:0 port:0) ---

         board: 0
          port: 0
    polling ia: 0x24
NumOfPollingTas: 0
      HostCode: 0x00
```

```
         Flags: 0x2f  (20:MSGSEG 08:EXCHARS 04:CTSREQ 02:DCDREQ 01:DSRREQ )
        Flags1: 0x29  (20:SEGSPOLL 08:PROTALLOW 01:APITHROT )
   NumOfDataIas: 0x01
0x24
     maxsegsize: 0098
    maxsegspoll: 0030
    txwaitcount: 0
--- Configuration Loaded ---

C (445:130) 10:13:05:00 hostio.c:1358         {----} Onboard version 3.8
C (445:131) 10:13:05:01 server.c:386          {----} Changed file descriptor limit from 64 to 1024
C (445:130) 10:13:05:01 hostio.c:574          {----} Receiving on device:</dev/innosb0> port:<alc_board_0> fd:5
C (445:130) 10:13:05:01 alcdata.c:924         {----} Host status: hostup:0x1  linkstat:0x7  miscstat:0x0  IA:0x24
       </dev/innosb0> <alc_board_0>
```

Note the “Host Status” in the last line. The “hostup:0x1” and the “linkstat:0x7” indicate that the gateway is properly started and that it is being polled by the host on at least one of the IA’s that are configured.

If the gateway starts and the host is polling but none of the IA’s that the gateway is configured for are being polled, then “hostup” will be “0x0” instead of “0x1”.

If the gateway starts and the host is not polling or is not properly cabled to the gateway, then the “Host status” line will not appear.

Note: Linkstat has the following possible values:

0x7 mean all modem signals are good
0x6 means DCD is missing
0x5 means DSR is missing
0x4 means DCD and DSR are missing
0x3 means RTS/CTS handshaking is missing
0x2 means RTS/CTS handshaking and DCD are missing
0x1 means RTS/CTS handshaking and DSR are missing
0x0 means no modem control signals are present

Hostup has the following possible values:

0x1 means one of the IA’s that the gateway is configured for is being polled
0x0 means none of the IA’s that the gateway is configured for are being polled

Appendix F — Typical Gateway startup under X.25

The following text is the expected output from the X.25 gateway (x25gate) when it is started with the debugging set to -vff. The lines that begin “*** Alarm” should be ignored. Following the X.25 gateway startup is the expected output from the ALC gateway (iate_server) when it is connecting to an X.25 gateway.

```
  InnoSys INSCC-Sbus X.25 Gateway
  x25gate ver 2.3.1  as of Aug 29 1996 15:48:48
Version control file /etc/innover missing;
should contain x25gate 2.3 to run this software
Reading configuration from <x25cfg0.sabre>
SVC 4 6
PVC 1 3
Alternate TCP service [x25gate0]
FLAG 1
CTS 1
DCD 1
DSR 1
hostname is <ultra1>  IP address is 207.88.91.61  port is 1414
Load and start board device </dev/innosb0>  with <x25.bin>...
INSCC Reset after 1
confgbuf:
  mem size f000  HW Vers 200  HW check 0  HW type 4
  clk speed c8  maxlen 131  maxmsgs 12c  maxevents 800  logdatamax 305  error type 0
  testing RAM...  cs = fe40  off = 125  done after 8 secs
testing ROM checksum...  cs = fe40  off = 2d2  done after 1 secs
testing SCC...  cs = fe40  off = 31f  done after 0 secs
testing Ctl Reg...  cs = fe40  off = 43d  done after 1 secs
Test Result: 0  Mem size Tested f000
MAXLEN is 305
Onboard initialization done after 1
Onboard SYSTEM_OK done after 1
confgbuf:
  mem size f000  HW Vers 200  HW check 0  HW type 4
  clk speed c8  maxlen 131  maxmsgs 12c  maxevents 800 logdatamax 305 error type 0
  eventlog 12acc  to_brd 14adc  to_gate 14c28
Downloading config...
12:48:54  nonDataToBoard 0  dataToBoard 0  fromBoard 0  fromClients 0
(cor chn fre len off)      BOARD   <--->  GATEWAY   <--->  CLIENT
(  0   0 499  96   0)                   <--- LINKINIT
Starting server...
boardFD 8  tcpSoc 6 ticostordSoc 7
POLL 1:  FD 8  e 1  tlook ffffffff
(  0   0 499   1   0)                   <--- HEARTBEAT
POLL 1:  FD 8  e 1  tlook ffffffff
(  0   0 499   2   0)    RESTARTPENDING --->
*** Alarm 103, 13  Severity 2  LCN 0  Restart Pending
(  0   0 499   2   0)                   <--- RESTARTALLOWED
POLL 1:  FD 8  e 1  tlook ffffffff
POLL 1:  FD 8  e 1  tlook ffffffff
12:48:55  nonDataToBoard 0  dataToBoard 0  fromBoard 0  fromClients 0
```

```
(21845   0 499   0   2)     RESTARTCOMPLETE --->
*** Alarm 104, 13  Severity 2  LCN 0  Restart Complete
POLL 1:  FD 8  e 1  tlook ffffffff
12:49:02  nonDataToBoard 0  dataToBoard 0  fromBoard 0  fromClients 0
(  0   0 499   8   0)       HEARTBEAT --->
INSCC freePool 2651/2653  freeMsg 300/301
POLL 1:  FD 6  e 1  tlook 1
New Client <0>, socket <9>                                (Note - ALC gateway
    connecting)
POLL 1:  FD 9  e 1  tlook 4
12:49:09  nonDataToBoard 0  dataToBoard 0  fromBoard 0  fromClients 0
(  0   0 499 298   0)                      <--- CALL
SVC Call: from   to 9188328410  user len 4: ffffffc1 00 00 00
(  0 65535 499 298   0)   <--- CALL
POLL 1:  FD 8  e 1  tlook ffffffff
POLL 1:  FD 8  e 1  tlook ffffffff
12:49:10  nonDataToBoard 0  dataToBoard 0  fromBoard 0  fromClients 0
(  0   6 499 298   0)       ACCEPT --->
(  0   6 499 298   0)                      ACCEPT --->
POLL 1:  FD 9  e 1  tlook 4
(  0   0 499  39   7)                                       <--- DATA
(  0   6 499  39   7)                      <--- DATA
POLL 1:  FD 8  e 1  tlook ffffffff
POLL 1:  FD 8  e 1  tlook ffffffff
(  0   6 499   0   0)       XON --->
(  0   6 499   0   0)                      XON --->
POLL 1:  FD 8  e 1  tlook ffffffff
12:49:11  nonDataToBoard 0  dataToBoard 0  fromBoard 0  fromClients 0
(  0   6 499   9   5)       DATA --->
(  0   6 499   9   5)                      DATA --->
(  0   6 499   0   0)                      <--- XON
POLL 1:  FD 8  e 1  tlook ffffffff
POLL 1:  FD 9  e 1  tlook 4
(  0   0 499   0   0)                                       <--- XON
POLL 1:  FD 8  e 1  tlook ffffffff
```

The following text is the expected output from the ALC gateway (iate_server) when it is connecting to an X.25 gateway and it is started with the debugging set to -vff. Each line starts with a type code (C=Comment, W=Warning, E=Error); queue counts (nn:nn); and a time stamp (dd:hh:mm:ss).

```
@(#)InnoSys IATE_SERVER Version 2.3b.5 (solaris) as of Jun 10 1997 12:48:07

This Gateway is licensed as follows:

  7 SABRE connections

C (17:-1) 10:12:49:09 debug.c:175         {----} Setting Gateway debugging to 0x00ff
C (17:-1) 10:12:49:09 vport.c:339         {----} vport tables has room for 1024 entries
C (17:-1) 10:12:49:09 socket.c:92         {----} Socket table has room for 1024 entries
C (17:-1) 10:12:49:09 config.c:620        {----} Processing configuration file scfg.x25.sabre.
C (17:-1) 10:12:49:09 config.c:3195       {----} Validating configuration file scfg.x25.sabre.
E (17:-1) 10:12:49:09 config.c:3257       {----} Unrecognized option <-TA_POLLING_ENABLED> at line 21 in
    file <scfg.x25.sabre>
```

```
C (17:-1) 10:12:49:09 vport.c:127          {----} Assigned vport 0 for use by socket type 1:HOST
C (17:-1) 10:12:49:09 config.c:675         {----} Configuring line for protocol SABRE.
C (22:-1) 10:12:49:09 config.c:698         {----} Configuring X25 (gateway) connection on "board":20 port:0
    vport:0.
C (22:-1) 10:12:49:09 config.c:702         {----} Using gateway/service <x25gate0>.
C (22:-1) 10:12:49:09 config.c:703         {----} Configuring line for pad type SABRE.
Setting CUD for port <0:x25_port_0> len:4: c1 00 00 00
C (71:-1) 10:12:49:09 config.c:770         {----} Line configured for 1 ias, 7 tas, and 0 polling tas.
C (71:-1) 10:12:49:09 vport.c:591          {----} Added vport 0:<x25_port_0> to x25_list

This Gateway has objects configured for the following host types:

  7 SABRE objects

A total of 7 objects are defined.

C (71:-1) 10:12:49:09 qhigh.c:150          {----} Allocated storage for 71 queues of 52 bytes each. (total 3692)
C (71:-1) 10:12:49:09 qhigh.c:181          {----} Allocated storage for 24 queue_items of 2316 bytes each.
    (total 164436)
C (39:24) 10:12:49:09 nettli.c:418         {----} Binding ialcserver at port 1413 on host ultra1, ip address
    INADDR_ANY, to socket 4.
C (39:24) 10:12:49:09 socket.c:275         {----} Initializing socket record:4 type:5:LISTEN_SOCK
C (36:24) 10:12:49:09 socket.c:275         {----} Initializing socket record:5 type:2:HOST_GATE
C (36:24) 10:12:49:09 hostio.c:284         {----} vport:0 <x25_port_0> opened:x25gate0 as:5

--- Opening connection with X25 Gateway (gate:x25gate0 vport:0) ---
C (36:24) 10:12:49:09 x25cmn.c:1103        {----} State change for <x25_port_0> vport:0 0:IXNULL-
    >1:IXSESSION
C (36:23) 10:12:49:09 x25sabre.c:959       {----} State change for <x25_port_0> vport:0 1:IXSESSION-
    >2:IXCALLED
C (36:23) 10:12:49:09 x25sabre.c:982       {----} CALLing <9188328410> <> <>
C (36:23) 10:12:49:09 route.c:670          {----} Routing X25 message on <x25_port_0> vport:0
    cmmd:0x31:CALL len:298
C (36:23) 10:12:49:09 hostio.c:998         {----} Sending on <x25_port_0> fd:5 vport:0 cmmd:0x31:CALL
    (298:12:310)
C (36:24) 10:12:49:10 server.c:386         {----} Changed file descriptor limit from 64 to 1024
0000: poll (POLLIN) completed on fd 5 <HOST_GATE>
C (36:23) 10:12:49:10 hostio.c:779         {----} Receiving on <x25_port_0> fd:5 vport:0 cmmd:0x32:ACCEPT
C (36:23) 10:12:49:10 x25cmn.c:107         {----} Processing message from <x25gate0> <x25_port_0> vport:0
    cmmd:0x32:ACCEPT
C (36:23) 10:12:49:10 x25sabre.c:130       {----} State change for <x25_port_0> vport:0 2:IXCALLED-
    >3:IXSVC
C (36:23) 10:12:49:10 route.c:670          {----} Routing X25 message on <x25_port_0> vport:0
    cmmd:0x3b:DATA len:39
C (36:23) 10:12:49:10 x25cmn.c:776         {----} Routing (immediate) on <x25_port_0> vport:0
    cmmd:0x3b:DATA len:39
C (36:23) 10:12:49:10 hostio.c:998         {----} Sending on <x25_port_0> fd:5 vport:0 cmmd:0x3b:DATA
    (39:12:51)
0000: poll (POLLIN) completed on fd 5 <HOST_GATE>
C (36:23) 10:12:49:10 hostio.c:779         {----} Receiving on <x25_port_0> fd:5 vport:0 cmmd:0x3f:XON
C (36:23) 10:12:49:10 x25cmn.c:107         {----} Processing message from <x25gate0> <x25_port_0> vport:0
    cmmd:0x3f:XON
0000: poll (POLLIN) completed on fd 5 <HOST_GATE>
C (36:23) 10:12:49:11 hostio.c:779         {----} Receiving on <x25_port_0> fd:5 vport:0 cmmd:0x3b:DATA
    (q_bit)
```

```
C (36:23) 10:12:49:11 x25cmn.c:107        {----} Processing message from <x25gate0> <x25_port_0> vport:0
    cmmd:0x3b:DATA
C (36:23) 10:12:49:11 x25sabre.c:775      {----} State change for <x25_port_0> vport:0 3:IXSVC->8:IXDATA
C (36:23) 10:12:49:11 x25cmn.c:1043       {----} Setting host status:0x701
C (36:22) 10:12:49:11 alcdata.c:924       {----} Host status: hostup:0x1  linkstat:0x7  miscstat:0x0  IA:0x12
    <x25gate0> <x25_port_0>
C (36:23) 10:12:49:11 route.c:670         {----} Routing X25 message on <x25_port_0> vport:0
    cmmd:0x3f:XON len:0
C (36:23) 10:12:49:11 hostio.c:998        {----} Sending on <x25_port_0> fd:5 vport:0 cmmd:0x3f:XON
    (0:12:12)
```

Note the “Host Status” in the third line up from the bottom. The “hostup:0x1” and the “linkstat:0x7” indicate that the ALC Gateway is properly started and that it is communicating with the airline host through the X.25 gateway.

Note: Linkstat has the following possible values:

0x7 mean all modem signals are good
0x6 means DCD is missing
0x5 means DSR is missing
0x4 means DCD and DSR are missing
0x3 means RTS/CTS handshaking is missing
0x2 means RTS/CTS handshaking and DCD are missing
0x1 means RTS/CTS handshaking and DSR are missing
0x0 means no modem control signals are present

Appendix G — X.25 PAD Type Specifications

This section describes the characteristics of the various PAD types supported by the Sun IATE Gateway. If the proper PAD type for an X.25 connection is not already known, the following information should help determine which PAD type to use.

Sabre PAD Type

- used only for the SABRE host type
- PVCs and SVCs are allowed
- Q_BIT messages are supported
- M_BIT messages are supported
- One host (X.121 or “Hunt Group”) address is required
- One local address is optional
- Call User Data is six bytes: the first four bytes, c1 00 00 00, are configured in the gateway CUD field, then the gateway automatically appends the configured LNIA to these four bytes.
- Call User Data is allowed in the call
- Call User Data is allowed in the ACCEPT
- The default behavior is for the gateway to make a call when it starts up
- The character set is EBCDIC
- Each data message has a three-byte header. The format of the header is:

byte 1	line number
byte 2	IA (in octal)
byte 3	TA

(the header is ALC, not EBCDIC)

- Either one IA per virtual circuit (one IA per config file) or multiple IAs per virtual circuit (multiple IAs per config file)

IATA PVC PAD Type (for Air New Zealand)

- PVCs only are supported
- Q_BIT messages are not used
- M_BIT messages are supported
- Host and local addresses do not apply
- The character set is padded ALC
- Each data message has a three-byte header. The format of the header is:

byte 1	line number
byte 2	0x00
byte 3	TA

IATA SVC PAD Type (for Worldspan)

- This PAD type is also called the IATA PAD type.
- SVCs only are supported
- Q_BIT messages are not used
- M_BIT messages are supported
- One host (X.121) address is required, up to 3 may be configured
- Local address is required
- Call User Data is nine bytes: fe 12 10 00 a1 00 00 00 00
- Call User Data is allowed in the call

- Call User Data is not allowed in the ACCEPT
- The default behavior is for the gateway to not make a call when it starts up
- The character set is padded (Reverse inverted) ALC
- Each data message has a three-byte header. The format of the header is:

byte 1	line number (hex)
byte 2	IA (Reverse Inverted PARS encoding)
byte 3	TA (Reverse Inverted PARS encoding or Native + 40 encoding)

- One IA per virtual circuit (one IA per config file), for multiple IAs use multiple VCs.
- Calls are only accepted if both:
  1) The called address matches the local X.121 number, and
  2) The calling address matches one of the three host X.121 numbers

IATA SVC PAD Type (for Galileo)

- This PAD type is also called both the EMTOX and the ASTOX PAD type.
- SVCs only are supported
- Q_BIT messages are not used
- M_BIT messages are supported
- One host (X.121) address is required, up to 3 may be configured
- Local address is required
- Call User Data is typically sixteen bytes: fe 12 11 00 91 00 00 00 00 00 00 00 00 01 00 00 (note: the call user data required for Galileo is often different than the default value provided by the Gateway)
- Call User Data is allowed in the call
- Call User Data is not allowed in the ACCEPT
- The default behavior is for the gateway to make a call when it starts up
- The character set is padded (Reverse inverted) ALC
- Each data message has a three-byte header. The format of the header is:

byte 1	line number 1 (hex)
byte 2	line number 2 (hex)
byte 3	TA (Reverse Inverted PARS encoding or Native + 40 encoding)

- One IA per virtual circuit (one IA per config file), for multiple IAs use multiple VCs.

Apollo PAD Type

- Used only for the Apollo host type
- IAs and TAs are configured by the user but not sent to the host
- SVCs only are supported
- Q_BIT messages are not used
- M_BIT messages are supported
- One host (X.121) address is required
- One local address is required and up to three may be used
- Call User Data is typically four bytes: d3 54 57 4e. Up to 16 bytes are allowed
- Call User Data is allowed in the call
- Call User Data is not allowed in the ACCEPT
- The default behavior is for the gateway to make a call when it starts up
- The character set is ASCII
- The no-op character is 0x00 instead of 0x10 and no-ops are stripped at the gateway
- Each data message has a twelve-byte header. The format of the header is:

byte 1-8	TID (Terminal ID)	
byte 9	Code set	=0x31 (ASCII)
byte 10	Data format	=0x31 (2915 terminal) or 0x32 (2915 printer)
byte 11	Status code	=0x30 (good)
byte 12	Reserved	=0x00
byte 13	C1	

byte 14	C2
byte 15	data message

- One IA per virtual circuit (one IA per config file), for multiple IAs use multiple VCs.
- On messages coming from the host, the C1 and C2 characters are represented as follows:

C1	is in ALC
C2	is in ALC, if for lines 1 through 12 is in ALC, modulo 0x38, if for lines 13 through 15

Appendix H — Gateway <--> Workstation Connectivity Problems

If a Windows workstation won't connect to the gateway, the following procedure can help isolate the problem. For other types of workstations, follow this same procedure but adapt it for the specifics of the workstation operating system.

1) Reboot the workstation PC and try to link again.

2) If the workstation says that it cannot find the gateway or the host is unreachable, this is probably because the "hosts" file is not set up properly or the dll has not been properly installed. Use the following procedure:

i) Try to "ping" the gateway using the tcp/ip address of the gateway PC. (The ping program should be on the workstation already). **If this is not successful, do not proceed until the gateway can be successfully pinged this way**. This kind of a problem is most likely caused by:

a) the installation of the tcp/ip protocol stack on the workstation machine is not correct/complete, or
b) there is not a network connection between the workstation and the gateway, or
c) the address being used for the gateway is not correct, or
d) there is a duplicate tcp/ip address on the network.

To "ping" the gateway using its tcp/ip address, enter a command such as:

```
ping 206.21.97.14
```

ii) If the gateway can be "pinged" using the tcp/ip address of the gateway PC, try to "ping" the gateway using the "host name" of the gateway PC. If this is not successful, the error is probably that:

a) the host name in the "hosts" file (or DNS) does not have the correct tcp/ip address for the gateway, or
b) the "hosts" file is in the wrong directory, or
c) the "hosts" file has an incorrect name (the most common error of this type is that the "hosts" file has an extension such as "sam". Use DOS, **not** Windows, to check the name of the "hosts" file because the Windows tools often hide the extension name.) The "hosts" file should **not** have an extension.

To "ping" the gateway using its host name, enter a command such as:

```
ping iate_gw1
```

The location of the "hosts" file varies according to which tcp/ip protocol stack is being used on the workstation. If the standard Microsoft tcp/ip protocol stack is being used under Windows 3.1, Windows for Workgroups, or Windows 95, the "hosts" file should be in the \WINDOWS directory. If the standard Microsoft tcp/ip protocol stack is being used under Windows NT, the "hosts" file should be in the \WINNT40\SYSTEM32\DRIVERS\ETC directory.

If a non-standard tcp/ip protocol stack is being used, the proper directory varies. For example, the "hosts" file is usually in the \NFS directory when using the PC/NFS tcp/ip protocol stack. The "hosts" file is usually in the \TRUMPET directory when using the

Trumpet Winsock tcp/ip protocol stack.

The “hosts” file is simply a list of names that can be used instead of actual tcp/ip addresses. For example, if the gateway machine’s tcp/ip address is 207.21.97.14 and the name assigned to this address is “iate_gw1”, the “hosts” file should contain a line that looks like this:

```
207.21.97.14      iate_gw1
```

iii) If the gateway can be “pinged” using the “host name” of the gateway PC, verify that the “iatedll.dll” file is properly installed. If running Windows 3.1, Windows for Workgroups, or Windows 95, “iatedll.dll” should be in the \WINDOWS directory. If running Windows NT, “iatedll.dll” should be in the \WINNT40 directory. If “iatedll.dll” is not in the correct directory, move it to the correct directory and then reboot before trying to link to the gateway again.

iv) If the “iatedll.dll” file is properly installed, then the most likely reason that the gateway cannot be found is that the “host name” is not properly entered in the “Configure Link” selection in the File menu. Check and recheck this entry with the indicated entry in the “hosts” file. Next, try entering the actual tcp/ip address of the gateway PC into the “host name” field instead of entering the name from the “hosts” file. Another thing to try is to search for all files named “hosts”. If there is more than one file named “hosts”, it is likely that the wrong hosts file is being used. If connecting to a Macintosh gateway via TCP/IP, verify that the “Gateway Name” configured in the IATEtcp software matches the “Gateway Name” used on the target Macintosh gateway.

3) If the terminal gives the message “socket-level connect failed”, this usually means that the port number entry that the gateway is using is not properly set up in the “services” file on the workstation PC. The standard Service Name used for IATE gateways is “ialcserver”. The standard Port Number used for IATE gateways is 1413. The “services” file is usually located in the same directory as the “hosts” file. Check that this file has a proper entry for the IATE gateway being used. If the standard names are being used, there should be a line in the “services” file that looks like this:

```
ialcserver 1413/tcp
```

Verify that the service name is spelled properly and that “tcp” is not mistyped.

If there are routers, bridges, and/or firewalls on the network between the gateway and the workstation, verify that network traffic on the tcp/ip port number which the gateway is using can be passed through the routers/bridges/firewalls.

4) If the message “Another Terminal or API is using this Object Name” is received when the client software is started, another user or application is already using the object name (or group name) to which the client software is trying to link. Or, if the client software is running on a machine that has access to the Internet, it is possible that the client software may be trying to link to an IATE gateway somewhere on the Internet. For this reason, it is recommended that common airline-related names such as “KLM”, “Delta”, “JAL”, etc. are never used as the host name for the gateway machine because these names are also domain names on the Internet.

If the message “This Object Name could not be found at the gateway” is received, the client software has found the gateway but the object name (or group name) to which the client software is trying to link does not agree with any of the object names defined on the gateway.

««

Appendix I — Frequently Asked Questions about Encoding

Q: What is the “Encoding Feature”?
A: This feature scrambles messages sent over TCP/IP between the InnoSys IATE gateway and the IATE client API. This scrambling is intended only to discourage casual snooping of those messages on the network.

Q: Which versions have this feature?
A: Windows NT gateway:
    Gateway (iate_server) 2.2b.4 and above
  Windows workstations:
    api (iatedll.dll) 2.4b.2 and above
    api (iate32.dll) 2.4b.1 and above
  Unix:
    gateway (iate_server) 2.3b.1 and above
    api (iatelib.a) 2.4b.1 and above
  Mac:
    IATEtcp: 3.4.4b7 and above
    initAPI: 3.4.4b31 and above

Q: How is this different from “encryption”?
A: “Encoding” isn’t “encryption”. “Encryption” is intended for applications that require a higher level of security than that provided by this “encoding” feature. InnoSys makes absolutely no representations or warranties about how easy or hard it might be to ‘crack’ this encoding scheme. We believe it provides reasonable protection against casual snooping, but anyone who is determined to break it will be able to do so.

Q: How does it work?
A: The encoding algorithm is secret. Whatever shielding it provides relies on its being secret. The scrambling is context sensitive. There is a dependency on the data in the message. The nth character isn’t always scrambled in the same way.

Q: How does one turn this feature on and off?
A: It is necessary to explicitly turn it on in the Unix and NT gateways. It is automatically turned on by the Mac IATEtcp gateway. When it is turned on, it is automatically used with any client that connects to that gateway and that supports encoding.

Q: What about compatibility? Do I have to make sure both ends of the client/server connection have the ability to do this?
A: Backward and forward compatibility is preserved - the client and server negotiate to make sure both support encoding and it is only attempted if both sides can support it.

Q: What if I already have some other type of security scheme in use in my network?
A: No problem. It will all just work.

Q: Will this encoding effect the performance of my system?
A: No, encoding has no performance impact.

Q: What does InnoSys advise for customers who want stronger security than this ‘encoding’ feature provides?

A: InnoSys recommends that customers who require TCP/IP security beyond the minimal level provided by this “encoding” scheme use some other network security methodology, such as secure routers, etc. InnoSys does not represent itself as an expert in such network security issues and encourages those customers who have such concerns to contact the appropriate network security consultants or other professionals.

Appendix J — Sample Printer Config Files

A typical configuration file for printing to a serial printer is:

```
#
#       pcfg file for a TI810 configured for 4800 baud/even parity/7 data bits/1 stop bit
#

OBJECT_NAME             @klgw\ialcserver\hdcopy02
PORT_NAME               /dev/ttyb
BAUD                    4800
PARITY                  EVEN
CHARACTER_SIZE          7
STOP_BITS               1
USE                     SHARED
RTS_CTS
CR_TO_CRLF
```

A typical configuration file for printing to a disk file is:

```
#
#       pcfg file for printing to a disk file
#

OBJECT_NAME          @innogw\ialcserver\itin02
FILE_NAME            itin02.txt
COLLECT_ALL
```

A typical configuration file for printing to the spooler with a user-supplied command is:

```
#
#       pcfg file for using a spool command
#

OBJECT_NAME          @gatew1\ialcserver\itin02
SPOOL_HOST
SPOOL_CMD            lpr -Pprtr4 -#3
COLLECT_ALL
```

A typical configuration file for printing to a SABRE ATB2 printer is:

```
#
#       sample printer configuration file for a SABRE ATB2 printer
#

OBJECT_NAME             @aagw\ialcserver\tkt02
PRINTER_TYPE            SABRE_ATB2
```

```
DISABLE_FLOWCONTROL
ASSUME_ONLINE
AUTO_ANSWER             OFF
PORT_NAME               /dev/ttyb
BAUD                    4800
PARITY                  NONE
CHARACTER_SIZE          8
STOP_BITS               1
```

Appendix K — Changing Gateway Translations

Any of the host to workstation or workstation to host translations used by the iate_server software can be changed. This is done using the “-x” on the command line to point to a file which specifies alternate translations.

This special alternate translations file has three types of lines in it:
- lines that name the specific translate table, plus any two-way translations.
- lines that describe changes in host to terminal translations.
- lines that describe changes in terminal to host translations.

Translate table & two-way changes: Since a single gateway can support more than one host type, it is possible to change the translations for more than one host. The changes for each different table must be preceded by a line that specifies which translate table is to be changed. Even though there are over twenty host types that the iate_server software supports, there are only 7 translate tables. Therefore, changing a translate table may affect more than one host type. The name of the seven tables are sabre, apollo, pars, klm, korean, isea, and galileo.

Subsequent lines in this section have one translation change per line, with the hex value of the host ALC character followed by the hex value of the workstation ASCII translation. Translation changes in this section apply to both data going from the host to a workstation and data going from a workstation to the host.

Host to Terminal “one-direction” changes: This section starts with a line that contains the characters “h>t”. Subsequent lines in this section have one translation change per line, with the hex value of the ALC character received from the host followed by the hex value of the desired ASCII translation.

Terminal to Host “one-direction” changes: This section starts with a line that contains the characters “t>h”. Subsequent lines in this section have one translation change per line, with the hex value of the desired ALC character followed by the hex value of the ASCII character received from the workstation.

A character can be translated in only one direction by only including it in the h>t or t>h section. There is no requirement that the translation be the same going from the host to a workstation and from a workstation to the host.

The following file shows shows how to change the translate table for galileo so that
i) ALC 1b is translated to ascii 3c and ascii 3c is translated to ALC 1b.
ii) ALC 2f is translated to ascii 29.
iii) ALC 0e is translated to ascii 28.
iv) ASCII 2e is translated to ALC 0c.

```
galileo
1b 3c
h>t
2f 29
0e 28
t>h
0c 2e
```

Setting the 0x80 bit of an ALC character in this file causes the translation to be applied to the extended character translations. This only applies to the Sabre and the Korean host types.

IATE™

ALC and X.25 Gateway

Installation Manual and User's Guide
for Sun Systems

InnoSys
INCORPORATED
3095 Richmond Parkway, Suite 207
Richmond, CA 94806
+1 510 222-7717

NO WARRANTIES OF ANY KIND ARE EXTENDED BY THIS DOCUMENT. The information herein and the IATE™ products themselves are furnished only pursuant to and subject to the terms and conditions of a duly executed Product License. INNOSYS SPECIFICALLY DISCLAIMS ALL WARRANTIES, WHETHER IMPLIED OR EXPRESSED, INCLUDING BUT NOT LIMITED TO THOSE OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. InnoSys has no responsibility, financial or otherwise, for any result of the use of this document and/or the associated product, including direct, indirect, special and/or consequential damages. The information contained herein is subject to change without notice.

INSCC boards have been tested and found to comply with the limits for CE conformity; for Class B digital devices, pursuant to Part 15 of the FCC Rules; for the Japanese VCCI standards; and for similar standards. The FCC Class B approval is deemed to be satisfactory evidence of compliance with Canada’s ICES-003 of the Canadian Interference-Causing Equipment Regulations. All of these standards and limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If an INSCC board does cause harmful interference to radio television, or other reception, which can be determined by turning the computer off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient the receiving antenna
- Increase the separation between the computer and receiver (radio or TV)
- Connect the computer into an outlet on a circuit different from that to which the receiver is connected.

Shielded cables must be used with the INSCC boards to insure compliance with emission limits. Changes or modifications to the INSCC not expressly approved by InnoSys could void the customer’s right to operate the equipment.

CE Declaration of Conformity

According to EN 45014

Manufacturer’s Name and Address

InnoSys Incorporated
3095 Richmond Parkway, Suite 207
Richmond, CA 94806

Declares that the product:

Product Name:	INSCC
Model Number:	INSCC-S

Conforms to the following Product Specifications:

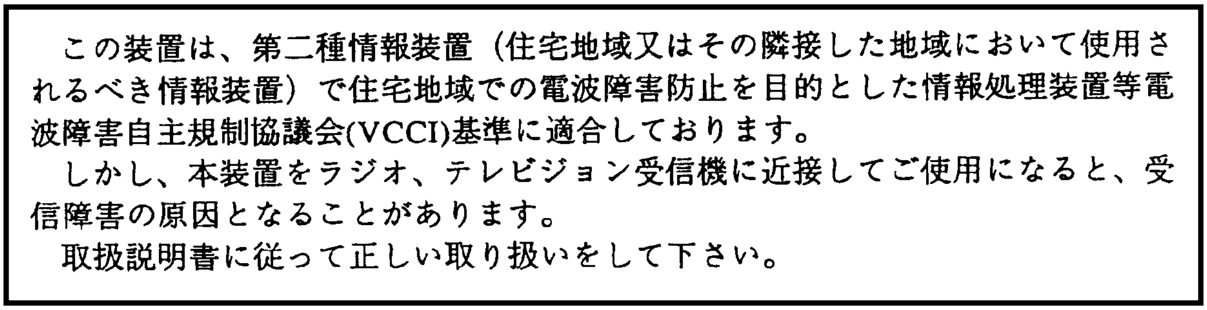
EMC: EN 55022: 1994 Class B
EN 50082-1:1992
IEC 801-2:1984 - 4kV CD, 8 kV AD
IEC 801-3:1984 - 3 V/m
IEC 801-4:1988 - 1 kV Power Lines, .5 kV Signal Lines

following the provisions of the Electromagnetic Compatibility Directive.

It also meets the EN60950:1992 standard, including amendments 1, 2 and 3, relating to the Low Voltage Directive (ITE).

Richmond, CA, USA Mike Ridenhour
June, 1996 President

Voluntary Control Council for Interference by
Information Technology Equipment



この装置は、第二種情報装置（住宅地域又はその隣接した地域において使用されるべき情報装置）で住宅地域での電波障害防止を目的とした情報処理装置等電波障害自主規制協議会(VCCI)基準に適合しております。

しかし、本装置をラジオ、テレビジョン受信機に近接してご使用になると、受信障害の原因となることがあります。

取扱説明書に従って正しい取り扱いをして下さい。

This equipment is in the 2nd Class category (information equipment to be used in a residential area or an adjacent area thereto) and conforms to the standards set by the Voluntary Control Council for Interference by Data Processing Equipment and Electronic Office Machines aimed at preventing radio interference in such residential area.

When used near a radio or TV receiver, it may become the cause of radio interference.
Read the instructions for correct handling.

InnoSys Incorporated
3095 Richmond Parkway, Suite 207
Richmond, CA 94806
(510) 222-7717 Voice
(510) 222-0323 FAX

««

Table of Contents

Introduction 1

System Requirements 1

ALC and X.25 Connections 2

Preparation 3

Getting the Gateway Running 4

Installation 5

Installing the S-Bus Card 5

Transferring files to the Machine 5

Installing the INSCC Device Driver and Making Special Device Nodes 7

Making the ALC Gateway known to the Network 9

Making the X.25 Gateway known to the Network 10

Configuration 11

Gateway Configuration 11

ALC Gateway Configuration 12

X.25 Gateway Configuration 23

Printer Configuration 26

Starting a Gateway or Printer 31

Starting and Stopping the X.25 Gateway 31

Loading the ALC Line Driver (INSCC-S Onboard Software) 32

Starting and Stopping the ALC Gateway 33

Using Shell Scripts to Start and Stop the Gateways 35

Starting and Stopping the Print Handler 35

Quick Start-Up (for an ALC-only configuration) 36

Supplemental Programs 37

serverwatch 37

testterm 37

showcfg 38

showdef 39

showxlat 39

showval 40

keyhelp 41

innostop 41

innoping 42

Board Test 43

Appendices 44

Appendix A: Files 44

Appendix B: Printer cabling 49

Appendix C: Modem Cabling 50

Appendix D: Sample Gateway Configuration files 51

Appendix E: Typical Gateway startup under ALC 58

Appendix F: Typical Gateway startup under X.25 60

Appendix G: X.25 PAD Type Specifications 64

Appendix H: Gateway <--> Workstation connectivity problems 67

Appendix I: Frequently Asked Questions about Encoding 69

Appendix J: Sample Printer Configuration files 71

Appendix K: Changing Gateway Translations 73