IRIS® 3270 Emulator Programming Guide

Document Number 007-1410-020

CONTRIBUTORS

Written by Sheri Chinen Biesen, Susan Thomas
Edited by Gail Larrick
Production by Julia Lin
Engineering contributions by Jay Lan, Fan Jiao, Robert Horen.

IRIS® 3270 Emulator Programming Guide
Document Number 007-1410-020

Contents

List of Figures

List of Tables

Introduction

The Silicon Graphics® IRIS® 3270 Emulator products provide a high-speed communications link between an IRIS-4D™ Series workstation and an IBM® host computer running either Virtual Machine/Conversational Monitor System (VM/CMS) or Multiple Virtual Storage/Time Sharing Option (MVS/TSO).

With the IRIS 3270 Emulator, you can use an IRIS-4D workstation or server—with Systems Network Architecture (SNA), Transmission Control Protocol (TCP), or 5080 Graphics System Workstation connectivity—to access your IBM mainframe. The list below shows how the various link types of the IRIS 3270 Emulator product connect to an IBM host.

IRIS 3270 Emulator Product:	Connects to IBM Host through:
SNA 3270 Emulator	IBM 3705, 3725, or 3745 front end processor
TCP 3270 Emulator	IBM 8232 Ethernet controller
3270 for the 5080 Emulator	IRIS Channel Adapter (ICA)

This document describes how to create High-Level Language Application Program Interface (HLLAPI) applications that interact with IBM host applications using Silicon Graphics' 3270 HLLAPI programming environment and tools. It is organized to provide programming information, including 3270 HLLAPI functionality and troubleshooting.

Chapter 1 explains Silicon Graphics' 3270 HLLAPI programming environment. Chapter 2 discusses troubleshooting using the trace display utility.

Note: For information on how to use and configure the IRIS 3270 Emulator, refer to the *IRIS 3270 Emulator User's Guid*e.

Typographical Conventions

These type conventions and symbols are used in this guide:

Italics	Filenames, variables, IRIX command arguments, command flags, titles of publications, icon names
Bold	Subroutine names
UPPERCASE	IBM file names, command names and the names of keys used on IBM systems
`Screen type`	Code examples, file excerpts, and screen displays (including error messages)
`Bold Screen typ`e	User input
()	(Parentheses) Following IRIX commands, they surround the reference page (man page) section where the command is described
[]	(Brackets) Surrounding optional syntax statement arguments
#	IRIX shell prompt for the superuser (*root*)

Software and Hardware

The IRIS 3270 Emulator software and hardware communicate between the IRIS workstation and an IBM-host system running VM/CMS or MVS/TSO. For information on system requirements, refer to the release notes included with your 3270 emulator product.

Software

The IRIS 3270 Emulator software includes these key files and directories:

/opt/3270/bin/	Contains all executable files for 3270
/opt/3270/chest/	Contains Toolchest menu files
/opt/3270/font/	Contains all fonts used by the emulator

/opt/3270/lib	Contains libsgi3270.a, the 3270 HLLAPI link library.
/var/opt/3270/example/	Contains examples of 3270 HLLAPI, IRISXFR and IND$FILE file transfer automation, file transfer input redirection, and keyboard input tracing.
/var/opt/3270/file/	Contains all log files used in problem determination
/var/opt/3270/spool/	Default location for files transferred to or from the host and screen captures
/var/opt/3270/lib/	Contains the 3270 configuration files
/usr/lib/X11/app-defaults/Setup3270	Contains color schemes for setup3270(1)

To use Silicon Graphics' IRISXFR file transfer utility, you must install the IRISXFR program on your IBM host. This software is distributed on a 1/2-inch tape, generated at 1600 bytes per inch (BPI). For information on how to install the host software, refer to the installation instructions included with the IRISXFR distribution.

Hardware

Warning: Do not attempt to add boards or other upgrades in your system. Hardware upgrades should be installed only by Silicon Graphics-certified personnel. Upgrades performed by noncertified persons void your warranty and may damage your system, or cause injury to improperly trained individuals.

Network Configuration

The IRIS workstation running 3270 emulator software can be connected to an IBM host through these network configurations:

- a leased line using Synchronous Data Link Control (SDLC) via an IBM 37X5 front end processor
- Ethernet, Token Ring, or FDDI using IBM 3172 or equivalent controller
- local non-SNA channel attach using the IRIS Channel Adapter (ICA)

The typical IRIS 3270 Emulator hardware configuration includes:

- an IBM, or IBM compatible, mainframe computer
- an IBM host front-end processor running Network Control Program (NCP)
- an IRIS SNA workstation running the IRIS 3270 Emulator software (SNA mode) through an IRIS SNA gateway
- an IRIS SNA gateway running the IRIS 3270 Emulator software (SNA mode)
- an IRIS Channel Adapter gateway and connections

Product Support

Silicon Graphics provides comprehensive technical hardware and software product support and a maintenance program for IRIS products. For more information, refer to the release notes that accompany this product.

Chapter 1

Silicon Graphics' 3270 HLLAPI

The Silicon Graphics 3270 high-level language application program interface (HLLAPI) provides a programming environment and tools to create 3270 HLLAPI applications that communicate with IBM mainframe applications. SNA, TCP, and 5080 modes are supported. The Silicon Graphics 3270 HLLAPI supports one session in SNA, TCP, and 5080 mode. Model 2, 3, 4, and 5 terminals are supported.

The Silicon Graphics 3270 HLLAPI also supports 3270 structured field data flows when using an SNA, TCP, or 5080 connection.

Silicon Graphics 3270 HLLAPI applications allow you to:

- create your own customized 3270 user interface for displaying and processing keyboard information
- automate repetitive sequences and run dialogues between the IRIS and IBM host unattended
- achieve faster throughput using structured field and Silicon Graphics Message Mode capabilities

The Silicon Graphics 3270 HLLAPI provides HLLAPI functionality equivalent to IBM 3270 PC application program interface (API) as specified in the*PC IBM 3270 Emulation Program, Version 3.00, Application Program Interface and Host Reference* manual, document version SC23-0960-0.

Input and return information is passed in a C structure rather than in registers. The values and meaning of returned codes are left unaltered. The number and type of the input structure members have also been maintained except for modifications introduced to support Silicon Graphics 3270 HLLAPI enhancements.

This chapter assumes that you understand the information contained in the *IRIS 3270 Emulator User's Guide* on using and configuring the IRIS 3270 Emulator.

IBM 3270 PC API to Silicon Graphics 3270 HLLAPI Changes

The following five changes map IBM PC Assembler implementation of HLLAPI to Silicon Graphics HLLAPI using C language:

- A one-to-one mapping between service requests and C subroutine calls eliminates the need to map the information in registers AH, AL, BH, BL, CX, and DX. This mapping also eliminates the need for the “system return code” that indicates improper values in these registers.
- The pointer to the parameter list (registers DX and ES) is replaced by a pointer to a structure.
- All instances of segment address/offset address in a parameter list are mapped onto pointers.
- All instances of “Must be zero/Unchanged” or “Reserved/Reserved” in the parameter list format definition are not implemented in the equivalent C structure.
- All elements of each C structure are aligned on 32-bit word boundaries. This protects the user against future changes (a byte field can be expanded to 2 bytes or a short word expanded to a long word).

Three subdirectories under */var/opt/3270/example* provide detailed examples of how to use the Silicon Graphics 3270 HLLAPI. All data structures used by the Silicon Graphics HLLAPI are defined in */usr/include/sys/hl_user_struct.h*. All constants used as input values or returned values are defined in */usr/include/sys/hl_user_define.h*. A short description of the contents of each subdirectory is provided below.

Case 1 provides an example of using **hl_entry_point** to create a filter routine within the Silicon Graphics-supplied *t3279* program. In this case, the keyboard and display processing is done by Silicon Graphics-supplied routines.

Case 2 is obsolete.

Case 3 provides an example of a 3270 emulation with prespecified input and no display processing. The screen descriptors supply the input. A comment file contains the output generated and processes the screen descriptors. To view this file, enter:

```
cat comment_output
```

You can also view a trace file by entering:

```
cp trace log /usr/3270/file
```

and then entering:

```
display_3270trace
```

Note: All code may be copied and used in your development effort.

Silicon Graphics 3270 HLLAPI Capabilities

The capabilities of Version 7.0 of the Silicon Graphics 3270 HLLAPI are listed here. The features listed in *italic* type are not provided by the IBM 3270-PC API.

- Determine the session ID
- Determine the value of the session parameters
- Determine the current cursor position
- Connect to the 3270 Presentation Space
- Disable keyboard input during HLLAPI activity
- Simulate keyboard data entry, that is, write data to the host application
- Enable keyboard input when HLLAPI activity is completed
- Disconnect from the 3270 Presentation Space
- Copy between the 3270 Presentation Space and an application data
- Read OIA group status
- Reinitialize Silicon Graphics 3270 HLLAPI after a UNIX® *exec* call while preserving the host session (for SNA, TCP, and 5080 modes only)
- Determine if the 3270 Presentation Space has been updated without a status line change
- Copy the status line from the 3270 Presentation Space into an application data area
- Start and stop Silicon Graphics Message Mode
- Read and write Silicon Graphics Message Mode Protocol Data Units

- Read and write 3270 structured fields
- Trace all data passing through the Silicon Graphics 3270 HLLAPI code

Subroutine Descriptions

Each description in this section explains the purpose of the subroutine, how it interacts with other Silicon Graphics 3270 HLLAPI subroutines, and the possible values for each element of the structure passed to the called routine. See */usr/include/sys/hl_user_struct.h* for the precise definition of each structure used by the service request subroutines. Appendix B lists the error messages that might be written to */var/opt/3270/file/hllapi_log* in the event that an error is detected.

Determine the Session ID

Calling Sequence	**void hl_query_session_id (struct session_id** **session_id_struct***)**
Purpose	Verify that the specified session is available.
Usage	Call this subroutine once for each desired session at the start of an application using the Silicon Graphics 3270 HLLAPI.

Input Structure Members

session_id	0x01—indicating a request to connect to Session 1
link_type	0x1—indicating a 5080 connection
	0x04—indicating SNA data over a leased line
	0x05—indicating a TCP/IP connection
lu_name	reserved

Returned Structure Members

session_ret	0x00—indicating success
	0x02—if Session ID is invalid
	0x2b—indicating a 3270 HLLAPI start up problem

	0x38—indicating that shared memory used by the Silicon Graphics HLLAPI could not be allocated
	0x39—indicating that the maximum number of sessions are already running
shmid	shared memory ID, used for *t3279*/*ps3279* communication

Initialize the Silicon Graphics 3270 HLLAPI Code

Calling Sequence

void hl_init (struct init **init_struct***)**

Purpose	Initialize the Silicon Graphics 3270 code at startup time.
Usage	Call this subroutine once, each time the Silicon Graphics 3270 HLLAPI application is started.

Input Structure Members

session_id	0x00—for first time initialization unique session number after an *exec* call command
command	0x01—indicates a first time initialization request
	0x02—indicates an initialization request after an *exec* call
config_name	full path name of configuration file
host_name	TCP/IP hostname for SGI gateway or IBM host
terminal_pool	LU name used only by SNA
signal_rtn	signal catching routine for SIGPOLL

Returned Structure Members

session_ret	0x00—indicating success

Determine the Value of the Session Parameters

Calling Sequence	**void hl_query_session_params (struct session_params** **session_params_struct***)**
Purpose	Determine the values of session characteristics that affect the operation of a Silicon Graphics 3270 HLLAPI application.
Usage	This subroutine can be called at any time after the Session ID has been determined. The Silicon Graphics 3270 HLLAPI deviates from the IBM 3270-PC API in that alternate screen sizes are supported. Extended Attributes are supported. Programmed Symbols are not supported.

Input Structure Members

session_id	unique session number

Returned Structure Members

session_type	0x02—is always returned
session_char	0x00—if no extended attributes and no programmed symbols
	0x40—if programmed symbols, but no extended attributes
	0x80—if extended attributes, but no programmed symbols
	0xc0—if extended attributes and programmed symbols
session_rows	number of rows in session’s Presentation Space
session_cols	number of columns in session’s Presentation Space
session_ret	0x00—if successful
	0x02—if Session ID is invalid

Determine the Current Cursor Position

Calling Sequence

void hl_query_session_cursor (struct session_cursor **session_cursor_struct***)**

Purpose	Determine the current cursor position and type.
Usage	This subroutine must be called whenever the Silicon Graphics 3270 HLLAPI application needs to determine the current cursor position and type. Row and column addresses start at 0, not 1.

Input Structure Members

session_id	unique session number

Returned Structure Members

cursor_type	0x00—if underscore cursor, always returned
cursor_row	current row address of cursor
cursor_col	current column address of cursor
cursor_pos	returned address, as offset
status	0x00—does not display cursor
	0x01—displays cursor
session_ret	0x00—if successful 0x02—if Session ID is invalid

Connect to a 3270 Presentation Space

Calling Sequence

void hl_connect (struct connect **connect_struct***)**

Purpose	Connect a Silicon Graphics 3270 HLLAPI application to an active 3270 Presentation Space.

Usage	This subroutine serializes access to keyboard and copy services. The routine must be called at the start of an Silicon Graphics 3270 HLLAPI application and normally is followed by a call to **hl_reserve**, which locks the keyboard, preventing operator input.

Input Structure Members

session_id	unique session number
query_reply	reserved
query_reply_len	reserved

Returned Structure Members

keyboard_ret	0x00—if successful
	0x02—if Session ID is invalid
	0x04—if session already connected for keyboard services

Disable Keyboard Input During HLLAPI Activity

Calling Sequence

void hl_reserve (struct reserve **reserve_struct***)**

Purpose	Causes 3270 emulator to disable keyboard input.
Usage	Normally, this subroutine is called immediately after **hl_connect** to prevent the intermingling of operator- and application-generated keyboard input.

Input Structure Members

session_id	unique session number

Returned Structure Members

keyboard_ret	0x00—if successful
	0x02—if Session ID is invalid
	0x04—if session not connected for keyboard services

Simulate Keyboard Data Entry

Calling Sequence

void hl_send_key (struct send_key **send_key_struct***)**

Purpose	Sends keystroke data from an Silicon Graphics 3270 HLLAPI application as an operator would enter it from a keyboard.
Usage	Used whenever the Silicon Graphics 3270 HLLAPI application wishes to send data to a host program. The Silicon Graphics 3270 HLLAPI deviates from the IBM specification by allowing an unlimited number of keystrokes to be encoded in the keystroke list.

Note: 3270 Buffer Code format is supported for all values of CHAR_SET. ASCII scan codes are supported only if CHAR_SET = US_ENGLISH in your configuration file.

n keystrokes

Scan Code #1	Shift State #1	. . . .	Scan Code #*n*	Shift State #*n*

Each scan code and shift state is an 8-bit quantity. See Appendix B, “Scan Code and Buffer Code Tables” for scan code shift state definitions.

Input Structure Members

session_id	unique session number
char_set	character set in use (possible values for *char_set* are defined in *hl_user_define.h*; they range from HL_FRENCH to HL_KATAKANA_ENGLISH)

Note: For Katakana applications, CHAR_SET = HL_KATAKANA implies that Katakana characters will be displayed. If English characters should be displayed, set CHAR_SET = HL_KATAKANA_ENGLISH.

keystroke_ptr	pointer to list of scan code/shift state pairs
total_keys	total number of keystrokes to be sent to the host

Returned Structure Members

keyboard_ret	0x00—if successful and no AID generated
	0x02—if Session ID is invalid
	0x04—if session not connected for keyboard services
	0x10—if invalid scan code or input inhibited condition
	0x12—if successful and AID generated
	0x37—if char_set invalid
keys_sent	number of keys sent before processing ended

Enable Keyboard Input When HLLAPI Activity Is Completed

Calling Sequence
void hl_release (struct release **release_struct***)**

Purpose Causes 3270 emulator to enable operator keyboard input.

Usage If **hl_reserve** has been called, **hl_release** can be called to reenable keyboard input. The **hl_disconnect** subroutine also reenables keyboard input automatically.

Input Structure Members

session_id	unique session number

Returned Structure Members

keyboard_ret	0x00—if successful
	0x02—if Session ID is invalid
	0x04—if session not connected for keyboard services

Disconnect from a 3270 Presentation Space

Calling Sequence

void hl_disconnect (struct disconnect **disconnect_struct***)**

Purpose	Disconnect a Silicon Graphics 3270 HLLAPI application from an active 3270 Presentation Space.
Usage	This subroutine serializes access to keyboard and copy string services. It must always be called prior to exiting the Silicon Graphics 3270 HLLAPI application code.

Input Structure Members

session_id	unique session number

Returned Structure Members

keyboard_ret	0x00—if successful
	0x02—if Session ID is invalid
	0x04—if session not connected for keyboard services

Copy Data between 3270 Presentation Space and Application Data Area

Calling Sequence

void hl_copy_data (struct copy_data **copy_data_struct***)**

Purpose	Copy part or all of the data from the 3270 Presentation Space to the Silicon Graphics 3270 HLLAPI application data area or vice versa.
Usage	Called whenever data must be moved from the 3270 Presentation Space to the application data area. It is also used to update the 3270 Presentation Space. Only the 3270 device buffer format is supported.

Input Structure Members

source_id	0x00—(application to 3270)
	unique session number (3270 to application)
source_ptr	0x00—(3270 to application)
	pointer to application data area (application to 3270)
source_beg	offset to first source character to be copied (both directions)
source_end	offset to last source character to be copied (both directions)
target_id	0x00—(3270 to application)
	unique session number (application to 3270)
target_ptr	0x00—(3270 to application)
	pointer to application data area (application to 3270)
target_beg	offset to starting copy position in target buffer (both directions)
copy_mode	0x00—3270 field attributes not copied (both directions)
	0x40—3270 field attributes copied (3270 to application)

Returned Structure Members

copy_ret	0x00—if successful
	0x02—if Session ID is invalid
	0x03—if target is input inhibited
	0x06—if invalid source definition
	0x07—if invalid target definition
	0x09—if truncation occurred
	0x0e—if some or all of the target is protected
	0x0f—if copy of field attributes not allowed

Read OIA Status Group

Calling Sequence	**void hl_read_oiag (struct read_oiag** **read_oiag_struct*)
Purpose	Determine the current input inhibited state.
Usage	This subroutine determines when the 3270 Presentation Space is ready to accept input. It is designed to be used in a polled mode or blocked mode with a timeout. Each unit of time represents 10 milliseconds.

Input Structure Members

session_id	unique session number
mode	0x01—if polling mode desired
	0x02—if blocked mode with timeout desired
timeout	1 - 6000—if *read_mode* = 2; 0 otherwise

Returned Structure Members

oiag_ret	0x00—if successful
	0x02—if Session ID is invalid
	0x23—if time out value not between 1 and 6000
oiag_buf	5 bytes containing returned OIAG status (see p. 1-1 to reference document SC23-0960-0, p. 3-73)

Determine if the Screen Has Been Updated

Calling Sequence	**void hl_screen_status (struct screen_status** **screen_status_struct*)
Purpose	Determine if any location on the screen has been updated.
Usage	This subroutine determines when the 3270 Presentation Space or Status Line has been updated. It eliminates the need to call **hl_read_oiag** and **hl_copy_data** to determine if any location on the screen has been updated. It also

provides a flag that indicates that a structured field has been received from the host and is waiting to be read by the user application. Each unit of time represents 10 milliseconds.

Input Structure Members

session_id	unique session number
mode	0x01—if polling mode desired
	0x02—if blocked mode with timeout desired
timeout	1 - 6000—if read_mode = 2; 0 otherwise

Returned Structure Members

screen_ret	0x00—if screen has been updated
	0x02—if Session ID is invalid
	0x23—if time out value not between 1 and 6000
	0x24—if the screen has not been updated
	0x25—if Unix has detected an error (error code is in errno)
line_change	44 bytes, one for each line, 0 = unchanged and 1 changed. line_change[0] represents the status line
sf_pending	1—if a structured field is waiting to be read, 0 otherwise
write_cmd	1—if screen updated by Erase Write or Erase Write Alternate command; 0 otherwise
partition	reserved
filexfr_active	reserved

Copy Status Line to an Application Data Area

Calling Sequence

void hl_copy_status (struct copy_status **copy_status_struct***)**

Purpose — Provides the Silicon Graphics 3270 HLLAPI application with the ability to display and/or analyze all fields of the 3270 status line.

Usage	Used in error recovery or when the terminal display program is customer-supplied. If HL_NO_COPY is specified, the *target_ptr* is not used and it is assumed that *stat_pointer* from the map data structure will be used to point to the status line. If HL_COPY_TO_APPL is used, *target_ptr* must be defined.

Input Structure Members

session_id	unique session number
mode	HL_NO_COPY and HL_COPY_TO_APPL
target_ptr	pointer to application data area (size >= 80 bytes)

Returned Structure Members

copy_ret	0x00—if the status line has been copied
	0x02—if Session ID is invalid

Start Silicon Graphics Message Mode

Calling Sequence

void hl_start_msg (struct start_msg **start_msg_struct***)**

Purpose	Allows the Silicon Graphics 3270 HLLAPI application to start Silicon Graphics Message Mode.
Usage	This subroutine must be invoked once prior to calling **hl_read_msg** or **hl_write_msg**. (CUT mode only)

Input Structure Members

session_id	unique session number

Returned Structure Members

message_ret	0x00—if successful
	0x02—if Session ID is invalid

Stop Silicon Graphics Message Mode

Calling Sequence

void hl_stop_msg (struct stop_msg **stop_msg_struct***)**

Purpose	Allows the Silicon Graphics 3270 HLLAPI application to terminate Silicon Graphics Message Mode.
Usage	Can be used to switch out of Silicon Graphics Message Mode. It need not be called when disconnecting, since **hl_disconnect** terminates Silicon Graphics Message Mode automatically.

Input Structure Members

session_id	unique session number

Returned Structure Members

message_ret	0x00—if successful
	0x02—if Session ID is invalid

Read Silicon Graphics Message Mode Protocol Data Units

Calling Sequence

void hl_read_msg (struct read_msg **read_msg_struct***)**

Purpose	Read a Silicon Graphics Message Mode Protocol Data Units sent by the host.
Usage	This subroutine is typically called when the application expects the host to send Message Mode data rather then normal 3270 data. All returned data is encoded in 3270 Buffer Code. It is assumed that each screen is written using the 3270 Erase Write command and that each screen in “unformatted”. Each Set Buffer Address (0,0) is marked by an 0xff character inserted into the data stream. If HL_NO_COPY is specified, the *data_ptr* is not used. Instead, the *mba_ptr* from the map data structure is used into point to the 64 Kb Message Mode ring buffer. The map data pointer is initialized by calling **hl_map_data**. The

application must manage its own offset into the ring buffer after initialization. If HL_COPY_TO_APPL is used, *data_ptr* must be defined.

Input Structure Members

session_id	unique session number
mode	HL_NO_COPY and HL_COPY_TO_APPL
data_ptr	pointer to application data area
data_size	size of the application data area

Returned Structure Members

message_ret	0x00—if successful
	0x02—if Session ID is invalid
	0x04—if session not connected for keyboard services
	0x21—if the receive time out value has been exceeded
	0x26—if the application data size < received data size
	0x28—if Silicon Graphics Message Mode has not been invoked
data_count	number of bytes stored in application data area

Read a 3270 Structured Field

Calling Sequence

void hl_read_sf (struct read_sf **read_sf_struct***)**

Purpose
Read a 3270 Structured Field that has been sent by the host.

Usage
This subroutine is called whenever an Silicon Graphics 3270 HLLAPI application expects a Structured Field from the host other than the Read Partition Query and Outbound 3270DS. The application need not call this routine unless it has already detected a structured field pending using the flag word provided by **hl_screen_status**. Each call to **hl_read_sf** returns the contents of a Structured Field. (SDLC)

Input Structure Members

session_id	unique session number
data_ptr	pointer to application data area
data_size	size of the application data area

Returned Structure Members

data_count	number of bytes stored in application data area
message_ret	0x00—if successful
	0x02—if Session ID is invalid
	0x04—if session not connected for keyboard services
	0x26—if the application data size < received data size

Write 3270 Structured Field to the Host

Calling Sequence

void hl_write_sf (struct write_sf **write_sf_struct***)**

Purpose: Send a 3270 Structured Field to the host.

Usage: This subroutine is called whenever the Silicon Graphics 3270 HLLAPI application must send a 3270 Structured Field to the host other than Query Reply and Inbound 3270DS. The Structured Field can contain up to 4K bytes of data. (DFT mode only)

Input Structure Members

session_id	unique session number
data_ptr	pointer to application data area
data_count	number of bytes stored in application data area

Returned Structure Members

message_ret	0x00 if successful
	0x02 if session ID is invalid
	0x04 if session not connected for keyboard services
	0x27 if the application data size > 64K bytes

Trace Silicon Graphics 3270 HLLAPI Data

Calling Sequence
void hl_trace (struct session_id **session_id_struct***)**

Purpose	Trace keyboard input and host output.
Usage	This routine is called whenever the Silicon Graphics 3270 HLLAPI application wants to trace all data passing through the Silicon Graphics 3270 HLLAPI. The trace data can be displayed by invoking */opt/3270/bin/display_3270trace*.

Input Structure Members

session_id	unique session number

Returned Structure Members

None

Get Silicon Graphics 3270 HLLAPI Information for a Specified Session

Calling Sequence
void hl_map_data (struct map_data **map_data_struct***)**

Purpose	Eliminate need to copy data from Silicon Graphics Presentation Space image into a local buffer provided by the Silicon Graphics 3270 HLLAPI application.
Usage	To minimize data copies and allow for different 3270 Presentation Space images, call once for each session at initialization time.

Input Structure Members

session_id	unique session number
ps_ptr	pointer to start of the 3270 Presentation Space
stat_ptr	pointer to start of the status line
mba_ptr	pointer to start of the Silicon Graphics Message Mode circular buffer
ea_ptr	pointer to start of the 3270 Extended Attribute Buffer
ps_rows_ptr	pointer to location containing number of rows
ps_cols_ptr	pointer to location containing number of columns

Returned Structure Members

session_ret	0x00—if successful

User Entry Point into the IRIS 3270 Emulator

Calling Sequence	
	void hl_entry_point (struct entry_point **entry_point_struct***)**
Purpose	Provide an entry point for user-supplied code enhancements to the IRIS 3270 Emulator.
Usage	This routine is called each time through the main processing loop for the IRIS 3270 Emulator. It is designed to provide an inline filter capability for user-specific needs allowing the filter routine to specify when the 3270 Presentation Space should be displayed for user viewing.

Input Structure Members

session_id	unique session number

Returned Structure Members

transfer_ret	0x35—Display the 3270 Presentation Space
	0x36—Do not display the 3270 Presentation Space
	0x3c—Poll 3270 Presentation Space for updates

Chapter 2

Troubleshooting

This chapter explains how to gather information and resolve or report problems that might appear when programming the IRIS 3270 Emulator.

The chapter includes information on using the trace utility to resolve HLLAPI errors.

Troubleshooting Using the Trace Display Utility

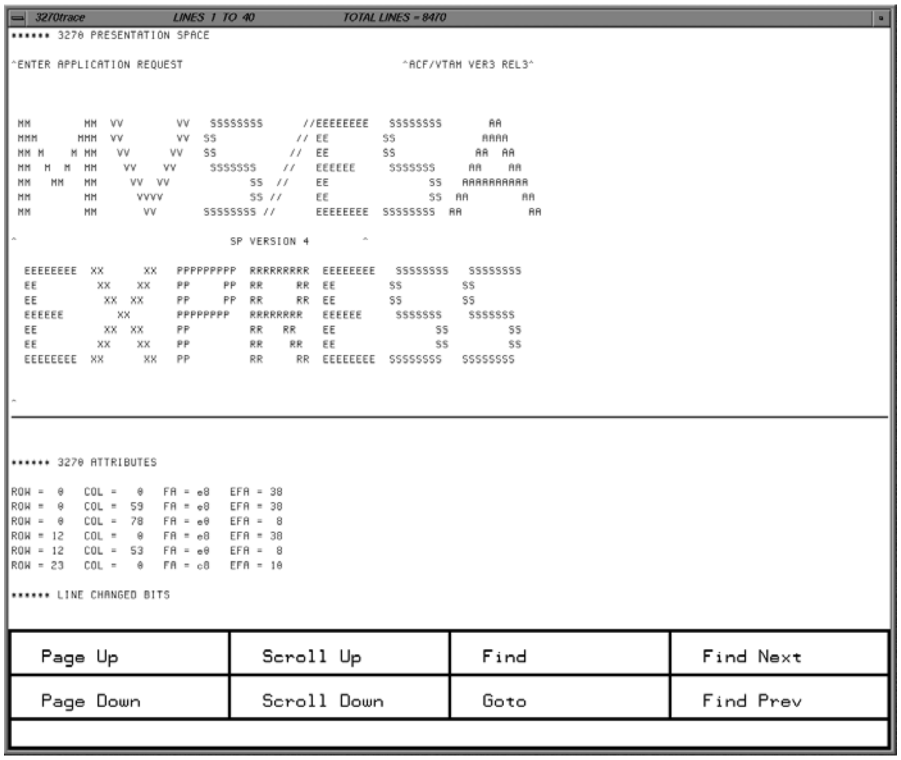
To track errors and more easily troubleshoot problems, you can use 3270 Trace to record all data sent to and received by the 3270 emulation process, then use the Trace Display utility to display the recorded data.

To record the presentation space data, turn on the 3270 Trace option from the *Miscellaneous* box in the Set Up 3270 window, as described in "Selecting Miscellaneous Options," in Chapter 1 of the *IRIS 3270 Emulator User's Guide*.

After you use 3270 Trace to record the presentation space, you can display the data using the Trace Display utility by entering:

```
display_3270trace
```

The full-screen window in Figure 2-1 appears.



```
****** 3270 PRESENTATION SPACE

^ENTER APPLICATION REQUEST                                 ^ACF/VTAM VER3 REL3^



 MM         MM  VV         VV   SSSSSSSS      //EEEEEEEE   SSSSSSSS       AA
 MMM       MMM  VV         VV  SS            // EE         SS            AAAA
 MM M     M MM   VV       VV   SS           //  EE         SS           AA  AA
 MM  M   M  MM    VV     VV     SSSSSSS    //   EEEEEE      SSSSSSS    AA    AA
 MM   MM    MM     VV   VV            SS  //    EE                SS  AAAAAAAAAA
 MM         MM      VVVV              SS //     EE                SS AA        AA
 MM         MM       VV        SSSSSSSS //      EEEEEEEE   SSSSSSSS AA          AA

^                                SP VERSION 4        ^

  EEEEEEEE  XX       XX   PPPPPPPPP  RRRRRRRRR  EEEEEEEE   SSSSSSSS   SSSSSSSS
  EE         XX     XX    PP      PP RR      RR EE        SS         SS
  EE          XX   XX     PP      PP RR      RR EE        SS         SS
  EEEEEE        XX        PPPPPPPP   RRRRRRRR   EEEEEE     SSSSSSS    SSSSSSS
  EE          XX   XX     PP         RR    RR   EE               SS         SS
  EE         XX     XX    PP         RR     RR  EE               SS         SS
  EEEEEEEE  XX       XX   PP         RR      RR EEEEEEEE  SSSSSSSS   SSSSSSSS


^
```

```
****** 3270 ATTRIBUTES

ROW =  0   COL =   0   FA = e8   EFA = 38
ROW =  0   COL =  59   FA = e8   EFA = 38
ROW =  0   COL =  78   FA = e0   EFA =  8
ROW = 12   COL =   0   FA = e8   EFA = 38
ROW = 12   COL =  53   FA = e0   EFA =  8
ROW = 23   COL =   0   FA = c8   EFA = 10

****** LINE CHANGED BITS
```

Figure 2-1 3270 Trace Display Window

Resolving Errors

This section deals with the process for resolving errors. Error messages are listed in Appendix A, "Silicon Graphics HLLAPI Error Messages."

Error Logging

Errors reported by the IRIS 3270 Emulator are logged in the file */var/opt/3270/file/t3279_log.PID*. All errors in this file are also displayed in an error message window. Errors reported by the SGI HLLAPI library are logged in */usr/adm/SYSLOG.* These errors are not visually displayed, since it is expected that each 3270 emulator coded on top of the SGI HLLAPI interface has its own style of user interface.

Error Information Sources

For configuration errors, the error information in the log files should be enough. For other errors, additional information can be gathered by setting the 3270 Trace option to *Yes* using the Set Up 3270 window as described in Chapter 1 of the *IRIS 3270 Emulator User's Guide*.

Configuration Errors

For configuration errors, look up the error message and make the appropriate correction using the Set Up 3270 window. For ASCII terminal users, see Appendix B of the *IRIS 3270 Emulator User's Guide* for more information on editing the configuration file.

If the error does not go away, refer to your release notes for product support information.

Nonconfiguration Errors

To find and correct errors that are not related to configuration problems, perform these steps:

1. Set the 3270 Trace option to *Yes* using the Set Up 3270 window.
2. Start the terminal emulator from the icon.
3. Re-create the problem.
4. Report the problem. Refer to your release notes for instructions on where to send the files below:
    - all executable files from your */opt/3270/bin* directory
    - */var/opt/3270/file/*_log** (all log files)
5. If the Silicon Graphics 3270 Emulator is terminated by program error, a core file is saved in */usr/var/opt/spool*. Send this file in addition to those mentioned in step 4.

6. If the problem occurs during a file transfer using IRISXFR, enter **`sgixfr versions`** while logged in to your host ID. Report the returned information along with the information collected in steps 1-3.

SGI HLLAPI Trace Display Interpretation

If you are a developer working with the SGI HLLAPI, use the Trace Display option in the 3270 icon to display the data contained in the trace file */var/opt/3270/file/trace_log*.

These types of information are traced: [BULLET LIST]

- An image of the 3270 Presentation Space. The color code is blue.
- An image of the 3270 Status Line. The color code is white.
- "Line Change Bits." 1 = changed and 0 = no change. The leftmost bit is for the status line. The other bits are for lines 1 to n with n being 24, 27, 32, or 43. The color code is yellow.
- A hexadecimal display of all attributes with their row and column position. Row and column numbers start at 0, not 1. The color code is red.
- Keyboard input. Use Appendix B to interpret the scan codes and shift state representation. The color code is green.
- A hexadecimal display of structured field data read by *hl_read_sf.c* and written by *hl_write_sf.c*. The color code for received data is blue and for transmitted data, yellow. The characters are in EBCDIC.
- A hex display of SGI Message Mode data read by *hl_read_msg.c* and written by *hl_write_msg.c*. The color code for received data is blue and for transmitted data, yellow. The characters are in 3270 buffer code.

Appendix A

Silicon Graphics HLLAPI Error Messages

The messages listed in this appendix reside in the file */usr/adm/SYSLOG.*

These errors are generated by the Silicon Graphics HLLAPI routines described in Chapter 1, "Silicon Graphics' 3270 HLLAPI." The messages are divided into two categories: system errors and configuration errors. All system errors should be reported immediately to your SGI service representative. All configuration errors can be eliminated or corrected by changing parameter values in */var/opt/3270/lib/t3279rc.$LOGNAME*.

System Errors

In the error list below, *N* is the UNIX error number as defined in the *intro* online man page. The messages are listed below:

If any of these errors occurs, refer to your release notes for product support information.

```
ERROR: Open of /dev/t3270c0 failed errno = N
ERROR: Download ioctl call failed  errno = N
ERROR: Open of /var/opt/3270/file/exec.info failed  errno = N
ERROR: Open of /usr/adm/SYSLOG failed  errno = N
ERROR: Open of /var/opt/3270/file/trace_log failed  errno = N
ERROR: Open of 3270 configuration file failed  errno = N
ERROR: Fstat call for /var/opt/3270/lib/t3279uc failed errno = N
ERROR: Malloc call failed  errno = N
ERROR: Mmap call failed  errno = N
ERROR: Read of /var/opt/3270/lib/t3279uc failed  errno = N
ERROR: SCR_UD ioctl call failed  errno = N
ERROR: SET_DID ioctl call failed  errno = N
```

Configuration Errors

If a configuration error occurs, see Appendix B in the *IRIS 3270 Emulator User's Guide* for the valid inputs for each configuration parameter. If changing the inputs does not solve the problem, refer to your release notes for product support information.

```
ERROR: RECV_TIMEOUT must be between 1000 and 65535
ERROR: SEND_TIMEOUT must be between 1 and 2047
ERROR: KEY_TIMEOUT must be between 1 and 31
ERROR: LINK missing or defined improperly
ERROR: CHAR_SET missing or improperly defined
ERROR: MODEL must be between 2 and 5
```

Errors Reported Using 3270trace Only

```
Could not get temporary space to read in trace file.
Error detected while reading /var/opt/3270/file/trace_log: errno = N.
Illegal character detected in line number - Please reenter.
Line number too large - Please reenter.
No data in the trace file.
Search string contains zero characters - Please reenter.
```

Appendix B

Scan Code and Buffer Code Tables

Table B-1 Scan Codes

Scancode[a]	Base	Alt
0B	Erase EOF	Erase Input
0C	Invalid	Attn
0D	Tab	Invalid
0F	PF2	Invalid
1F	PF4	Invalid
2F	PF6	Invalid
3F	PF8	Invalid
4F	PF10	Invalid
5A	CR	Invalid
5E	PF12	Invalid
5F	PF24	Invalid
6A	cursor right	Invalid
6E	PA2	Invalid
05	Invalid	Sysrq
06	Clear	Invalid
07	PF1	Invalid
08	PF13	Invalid
10	PF14	Invalid

Table B-1 (continued) Scan Codes

Scancode[a]	**Base**	**Alt**
11	reset	Invalid
17	PF3	Invalid
18	PF15	Invalid
20	PF16	Invalid
27	PF5	Invalid
28	PF17	Invalid
30	PF18	Invalid
37	PF7	Invalid
38	PF19	Invalid
40	PF20	Invalid
47	PF9	Invalid
48	PF21	Invalid
50	PF22	Invalid
56	PF11	Invalid
57	PF23	Invalid
58	Enter	Invalid
60	cursor down	Invalid
61	Backspace	Invalid
61	cursor left	Invalid
62	Invalid	home
63	cursor up	Invalid
64	Back Tab	Invalid
65	Insert	Invalid
66	Delete	Invalid

Table B-1 (continued) Scan Codes

Scancode[a]	Base	Alt
67	PA1	Invalid
70	Field Mark	Dev Cncl
71	Dup	Test
72	Ident	Invalid
73	Shift On	Shift Off

a. All scan codes other than those listed are Invalid. Invalid scan codes are handled as indicated in the Write Keystroke service.

Table B-2 Buffer Codes

Character Description		ASCII Hex	3270 Device Buffer Hex	Valid Across Languages
Null		00	00	y
		Invalid	01	n
		Invalid	02	n
		Invalid	03	n
		Invalid	04	n
		Invalid	05	n
		Invalid	06	n
		Invalid	07	n
Greater Than sign	>	3E	08	y
Less Than sign	<	3C	09	y
Left Bracket	[	5B	0A	n
Right Bracket	]	5D	0B	n
Right Parenthesis	)	29	0C	y

Table B-2 (continued) Buffer Codes

Character Description		ASCII Hex	3270 Device Buffer Hex	Valid Across Languages
Left Parenthesis	(	28	0D	y
Right Brace	}	7D	0E	n
Left Brace	{	7B	0F	n
Blank		20	10	y
Equal sign	=	3D	11	y
Apostrophe	'	27	12	y
Double Quote	"	22	13	n
Slash	/	2F	14	y
Reverse Slash	\	5C	15	n
Vertical Bar	|	B3	16	n
Broken Vertical Bar		7C	17	n
Question Mark	?	3F	18	y
Exclamation Mark	!	21	19	n
Dollar sign	$	24	1A	n
Cent Sign	¢	9B	1B	n
Lb. sign - Lire	£	9C	1C	n
Yen sign	¥	9D	1D	n
Peseta	•	9E	1E	n
Intl. Monetary sign	¤	0F	1F	n
0 (F0)	0	30	20	y
1 (F1)	1	31	21	21
2 (F2)	2	32	22	y
3 (F3)	3	33	23	y

Table B-2 (continued) Buffer Codes

Character Description		ASCII Hex	3270 Device Buffer Hex	Valid Across Languages
4 (F4)	4	34	24	y
5 (F5)	5	35	25	y
6 (F6)	6	36	26	y
7 (F7)	7	37	27	y
8 (F8)	8	38	28	y
9 (F9)	9	39	29	y
Beta	ß	F1	2A	n
Section Symbol	§	15	2B	n
Number sign	#	23	2C	n
At sign	@	40	2D	n
Percent sign	%	25	2E	y
Underscore	_	5F	2F	y
Ampersand	&	26	30	y
Hyphen	-	2D	31	y
Period	.	2E	32	y
Comma	,	2C	33	y
Colon	:	3A	34	y
Plus sign	+	2B	35	y
Not Symbol	¬	AA	36	n
Overbar		Invalid	37	n
Degree		Invalid	38	n
		Invalid	39	n
Circumflex	^	5E	3A	n

Table B-2 (continued) Buffer Codes

Character Description		ASCII Hex	3270 Device Buffer Hex	Valid Across Languages
Tilde	~	7E	3B	n
Diaeresis	¨	Invalid	3C	n
Grave Accent	`	60	3D	n
Acute Accent	´	27	3E	n
Cedilla	ç	Invalid	3F	n
a Grave Small	à	85	40	n
e Grave Small	è	8A	41	n
i Grave Small	ì	8D	42	n
o Grave Small	ò	95	43	n
u Grave Small	ù	97	44	n
a Tilde Small	ã	Invalid	45	n
o Tilde Small	õ	Invalid	46	n
y Diaeresis Small	ÿ	98	47	n
		Invalid	48	n
		Invalid	49	n
		Invalid	4A	n
		Invalid	4B	n
		Invalid	4C	n
		Invalid	4D	n
		Invalid	4E	n
		Invalid	4F	n
a Diaeresis Small	ä	84	50	n
e Diaeresis Small	ë	89	51	n

Table B-2 (continued) Buffer Codes

Character Description		ASCII Hex	3270 Device Buffer Hex	Valid Across Languages
i Diaeresis Small	ï	8B	52	n
o Diaeresis Small	ö	94	53	n
u Diaeresis Small	ü	81	54	n
a Circumflex Small	â	83	55	n
e Circumflex Small	ê	88	56	n
i Circumflex Small	î	8C	57	n
o Circumflex Small	ô	93	58	n
u Circumflex Small	û	96	59	n
a Acute Small	á	A0	5A	n
e Acute Small	é	82	5B	n
i Acute Small	í	A1	5C	n
o Acute Small	ó	A2	5D	n
u Acute Small	ú	A3	5E	n
n Tilde Small	ã	A4	5F	n
a Grave Capital		Invalid	60	n
e Grave Capital		Invalid	61	n
i Grave Capital		Invalid	62	n
o Grave Capital		Invalid	63	n
u Grave Capital		Invalid	64	n
a Tilde Capital		Invalid	65	n
e Tilde Capital		Invalid	66	n
		Invalid	68	n
		Invalid	69	n

Table B-2 (continued) Buffer Codes

Character Description		ASCII Hex	3270 Device Buffer Hex	Valid Across Languages
		Invalid	6A	n
		Invalid	6B	n
		Invalid	6C	n
		Invalid	6D	n
		Invalid	6E	n
		Invalid	6F	n
a Diaeresis Capital	Ä	8E	70	n
e Diaeresis Capital		Invalid	71	n
i Diaeresis Capital		Invalid	72	n
o Diaeresis Capital	Ö	99	73	n
u Diaeresis Capital	Ü	9A	74	n
a Circumflex Capital		Invalid	75	n
e Circumflex Capital		Invalid	76	n
i Circumflex Capital		Invalid	77	n
o Circumflex Capital		Invalid	78	n
u Circumflex Capital		Invalid	79	n
a Acute Capital		Invalid	7A	n
e Acute Capital	É	90	7B	n
i Acute Capital		Invalid	7C	n
o Acute Capital		Invalid	7D	n
u Acute Capital		Invalid	7E	n
n Tilde Capital	Ñ	A5	7F	n
a Small	a	61	80	y

Table B-2 (continued) Buffer Codes

Character Description		**ASCII Hex**	**3270 Device Buffer Hex**	**Valid Across Languages**
b Small	b	62	81	y
c Small	c	63	82	y
d Small	d	64	83	y
e Small	e	65	84	y
f Small	f	66	85	y
g Small	g	67	86	y
h Small	h	68	87	y
i Small	i	69	88	y
j Small	j	6A	89	y
k Small	k	6B	8A	y
l Small	l	6C	8B	y
m Small	m	6D	8C	y
n Small	n	6E	8D	y
o Small	o	6F	8E	y
p Small	p	70	8F	y
q Small	q	71	90	y
r Small	r	72	91	y
s Small	s	73	92	y
t Small	t	74	93	y
u Small	u	75	94	y
v Small	v	76	95	y
w Small	w	77	96	y
x Small	x	78	97	y

Table B-2 (continued) Buffer Codes

Character Description		ASCII Hex	3270 Device Buffer Hex	Valid Across Languages
y Small	y	79	98	y
z Small	z	7A	99	y
Diphthong Small	æ	91	9A	n
o Slash Small		Invalid	9B	n
a Overcircle Small		86	9C	n
c Cedilla Small		87	9D	n
		Invalid	9E	y
		Invalid	9F	y
a Capital	A	41	A0	y
b Capital	B	42	A1	y
c Capital	C	43	A2	y
d Capital	D	44	A3	y
e Capital	E	45	A4	y
f Capital	F	46	A5	y
g Capital	G	47	A6	y
h Capital	H	48	A7	y
i Capital	I	49	A8	y
j Capital	J	4A	A9	y
k Capital	K	4B	AA	y
l Capital	L	4C	AB	y
m Capital	M	4D	AC	y
n Capital	N	4E	AD	y
o Capital	O	4F	AE	y

Table B-2 (continued) Buffer Codes

Character Description		ASCII Hex	3270 Device Buffer Hex	Valid Across Languages
p Capital	P	50	AF	y
q Capital	Q	51	B0	y
r Capital	R	52	B1	y
s Capital	S	53	B2	y
t Capital	T	54	B3	y
u Capita;	U	55	B4	y
v Capital	V	56	B5	y
w Capital	W	57	B6	y
x Capital	X	58	B7	y
y Capital	Y	59	B8	y
z Capital	Z	5A	B9	y
Diphthong Capital	Æ	92	BA	n
o Slash Capital		Invalid	BB	n
a Overcircle Capital		8F	BC	n
c Cedilla Capital		80	BD	n
Semicolon	;	3B	BE	y
Asterisk	*	2A	BF	y

Index

Tell Us About This Manual

As a user of Silicon Graphics products, you can help us to better understand your needs and to improve the quality of our documentation.

Any information that you provide will be useful. Here is a list of suggested topics:

- General impression of the document
- Omission of material that you expected to find
- Technical errors
- Relevance of the material to the job you had to do
- Quality of the printing and binding

Please send the title and part number of the document with your comments. The part number for this document is 007-1410-020.

Thank you!

Three Ways to Reach Us

- To send your comments by **electronic mail**, use either of these addresses:
  - On the Internet: techpubs@sgi.com
  - For UUCP mail (through any backbone site): *[your_site]*!sgi!techpubs
- To **fax** your comments (or annotated copies of manual pages), use this fax number: 650-965-0964
- To send your comments by **traditional mail**, use this address:

  Technical Publications
  Silicon Graphics, Inc.
  2011 North Shoreline Boulevard, M/S 535
  Mountain View, California 94043-1389