XDS Native Command Set Function Library For SCSA & H.100 Boards

October 2005

American Tel-A-Systems, Inc.

251M024O ©

Contents

24 H.100 / H.110 MVIP-Compatibility Commands

25 Received Message Types

XDS SCSA / H.100 Native Function Library Reference Manual

American Tel-A-System, Inc.©
608-838-4194
4800 Curtin Drive, McFarland, WI 53558, USA
http://www.amtelco.com/
251M024O

Library & IOCTL Common Definitions

This page was intentionally left blank.

Listed in this section of the manual are commonly used function library variables and function return codes.

General Library return codes:

ILL_PORT	1	Illegal Port
ILL_SLOT	2	Illegal Timeslot
ILL_ARG	3	Illegal Argument
ILL_HANDLE	4	Illegal Conference Handle
ILL_ATTEN	5	Illegal Attenuation
ILL_THRESHOLD	6	Illegal Threshold
WRONG_QUERY	7	Wrong Query
WRONG_BOARD	8	Wrong Board
NO_UPDATE	9	DPRAM value hasn’t changed
ILL_CCA	10	Illegal Conference Handle
ILL_MODE	11	Illegal “mode” value
ILL_TYPE	12	Illegal “type” (BRI) value
ILL_FEATURE	13	Illegal “feature” (BRI) value
ILL_CAUSE	14	Illegal “cause” (BRI) value
ILL_REFERENCE	15	Illegal “reference” (BRI) value
ILL_PROGRESS	16	Illegal “progress” (BRI) value
ILL_SIGNAL	17	Illegal “signal” (BRI) value
ILL_PORT_TYPE	18	Illegal port type
ILL_ID	19	Illegal reference ID code
ILL_PATTERN	20	Illegal pattern value
ILL_STREAM	21	Illegal stream value
ILL_TONES	22	Illegal DTMF tones selected
ILL_OPTION	23	Illegal parameter options

Ioctl return codes:

SUCCESS	0	ioctl() call returned successfully

UNIX only defines:

XDS_MSG_AVAILABLE	0	message available
XDS_BOARD_NOT_PRESENT	1	XDS board not present
XDS_NO_MSG	1	no messages on queue
XDS_BOARD_NOT_RESPONDING	2	XDS board is not responding
XDS_DPRAM_READ_OFF	2	attempt to read at an offset before the beginning of the board
XDS_DPRAM_BAD_WRITE	2	attempt to write to first 256 bytes of board’s DPRAM
XDS_DPRAM_WRITE_LIMIT	3	attempt to write beyond the 2k limit
XDS_DPRAM_READ_LIMIT	3	attempt to read beyond the 2k limit
XDS_BAD_COMMAND	4	non-supported ioctl command

Common Error codes:

256	‘XMT’ command failed
257	‘RCV’ command failed
258	‘RCV_QUERY’ command failed
259	‘WRITE_DPRAM’ command fail
260	‘READ_DPRAM’ command
261	Invalid ioctl() command
262	‘XDS_RESET’ command failed
263	‘XDS_BOARD_ID’ command failed
264	‘XDS_GET_BUS_DEVICE_NUM’ command failed
265	‘XDS_QUEUE_USER_MSG’ command failed
266	invalid device handle was used
267	VOICE_SET_EVENT command failed
268	VOICE_RESOURCE_UNSET_EVENT command failed
269	XDS board’s xmit flag is set and application (library) is unable to send message to board
270	XDS library failed to create timer
271	XDS_GET_BOARD_INFO command failed
272	XDS_VOICE_RCV failed
273	XDS_VOICE_WRITE_DPRAM failed
274	XDS_VOICE_RESOURCE_WRITE_DPRAM_DATA failed
275	XDS_VOICE_RESOURCE_RCV failed
276	XDS_VOICE_RESOURCE_XMT failed
277	XDS_VOICE_RESOURCE_SET_EVENT failed
278	XDS_VOICE_RESOURCE_UNSET_EVENT failed
279	XDS_IOCTL_VOICE_RESOURCE_READ_DPRAM failed
280	XDS_VOICE_RESOURCE_WRITE_DPRAM_PARAMETERS failed
281	XDS_PLX_INT failed

Commonly used variable ranges:

board_number:

ISA	-	0 – 15
H.100	-	16 – 31
H.110	-	1 – 30

port:

MVIP/SCSA BRI (ISA)	-	0 – 11
H.100 BRI	-	0 – 15
H.110 BRI	-	0 – 31
U-interface BRI (All)	-	0 – 3
4 Port BRI (All)	-	0 – 3

b_channel:

MVIP/SCSA (ISA)	-	0 – 23
H.100	-	0 – 31
H.110	-	0 – 63
U-interface (All)	-	0 – 7
4 Port (All)	-	0 – 7
T1 PRI (8 port)	-	0 – 247
T1 PRI (4 port)	-	0 – 119

timeslot:

MVIP	-	0 – 511
SCSA	-	0 – 1023
H.100 bus (MVIP mode)	-	0 – 511
H.100 bus (SCSA mode)	-	0 – 1023
H.100 bus	-	0 – 4095
H.110 bus	-	0 – 4095
MC3 bus	-	0 – 4845

This page was intentionally left blank.

Generic XDS Functions

This page was intentionally left blank.

xds_id

int xds_id(board_number, idp);

unsigned char board_number;	a value indicating which board the command is for
struct xdsid id, *idp;	a structure to contain the return information

typedef struct xdsid
{

char board_number;	the board number
char id[5];	a character array containing the board ID string
char version[5];	a character array containing the firmware version number
int number_ports;	the number of ports on the board
unsigned char pci_bus_number;	board's PCI bus number
unsigned char pci_device_number;	board's PCI device number
unsigned long board_memory_size;	XDS board DPRAM size
unsigned long vr_board_memory_size;	XDS Voice Resource board Blackfin DPRAM size (voice resource boards only)
unsigned char vr_board_number_channels;	number of voice channels available (voice resource boards only)
unsigned long vr_board_voice_buffer_size;	size of voice buffers (voice resource boards only)

}

Applicable Boards
All XDS boards

Purpose
This function will return the two, three, or four character ID code (depending on the type of board), the firmware version number, and the number of ports on the board. This information is returned in a structure of the type **xdsid**.

Message Sent
None.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This function can be used to obtain information about the configuration of the system, what type of XDS board is associated with each device number, and the number of ports available on each board. It functions by directly reading the Dual-Ported Ram. For the BRI boards, the port number is the number of physical interfaces.

Only PCI H.100 XDS board will return a valid value for pci_bus_number, pci_device_number, and board_memory_size.

Only XDS boards with a voice resource module will return a valid value for vr_board_memory_size, vr_board_number_channels, and vr_board_voice_buffer_size.

Examples

xds_id(16, &idstruct);	returns with the ID information for board 16 in idstruct; if board number 16 was an XDS H.100 MC3 board the structure would contain:
idstruct.board_number = 16;	board number
idstruct.id = “PO3 ”;	board type “PO3 ” indicates that it is an H.100 MC3 board
idstruct.version = “012e”;	firmware version 1.2e
idstruct.pci_bus_number = 0;	PCI bus number 0
idstruct.pci_device_number = 1;	PCI device number 1
idstruct.board_memory_size= 8192;	XDS DPRAM is 8K
idstruct.vr_board_memory_size = 0;	N/A (non – voice resource board)
idstruct.vr_board_number_channels= 0;	N/A (non – voice resource board)
idstruct.vr_board_voice_buffer_size= 0;	N/A (non – voice resource board)

xds_id(17, &idstruct);	returns with the ID information for board 17 in idstruct; if board number 17 was an XDS H.100 Loop Start board the structure would contain:
idstruct.board_number = 16;	board number
idstruct.id = “PL ”;	board type “PL ” indicates that it is an H.100 MC3 board
idstruct.version = “004b”;	firmware version 004b
idstruct.pci_bus_number = 0;	PCI bus number 0
idstruct.pci_device_number = 2;	PCI device number 2
idstruct.board_memory_size= 8192;	XDS DPRAM is 8K
idstruct.vr_board_memory_size = 65536;	voice resource module DPRAM is 64K
idstruct.vr_board_number_channels= 32;	32 voice channels
idstruct.vr_board_voice_buffer_size= 1792;	voice buffer size is 1792 bytes

xds_msg_on

int xds_msg_on(board_number);

unsigned char board_number;	a value indicating which board the command is for

Applicable Boards
All XDS boards

Purpose
This function is used to enable messages from the XDS board. It should be used at the beginning of an application.

Message Sent
“IN”

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
If this function is not called, the XDS board will not send messages. Therefore, it should be part of the initialization of an application. It will cause the board to respond with an “IA” message.

Example

xds_msg_on(1);	sends an IN message to board 1 to enable messages

xds_msg_off

int xds_msg_off(board_number);

unsigned char board_number;	a value indicating which board the command is for

Applicable Boards
All XDS boards

Purpose
This function is used to disable messages from the XDS board. It should be used at the end of an application unless multiple applications are using the same boards.

Message Sent
“IF”

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
If this function is not called at the termination of an application, the XDS board will continue to send messages is response to state changes or error detection. This will cause interrupts and may lead to problems in communicating with the board. Therefore, before closing an application, this function should be called for each XDS board. However, if more than one application is running, only the last application to close should call this function as it prevents messages from being sent by the board. No response message will be sent from the XDS board.

Example

xds_msg_off(1);	sends an IF message to board 1 to disable messages

xds_msg_receive

int xds_msg_receive(rcvp);
struct rcvmsg rcv, *rcvp {

unsigned char board_number;	a value indicating which board the command is for
char msg_type;	a value denoting the message type
char msg_sub;	a value denoting the message subtype
int timeslot;	the timeslot, if any, involved in the particular message. if not timeslot specific, this value should be (-1)
int port_number;	the port number, if any, involved in the particular message. if not port specific, this value should be (-1)
unsigned char argument;	a generalized argument field for the message whose meaning is specific to the particular message type, it will not have meaning for all message types
char msg_str[32];	a character array to pass any message specific values that do not fit in the other members of the structure
unsigned short augRxLen;	length of Layer 3 message in the auxiliary mailbox
unsigned char augRxMesg[260];	body of Layer 3 message in the auxiliary mailbox

}

Purpose
This function returns a structure with information about messages sent from an XDS board. A single receive queue is used for all XDS boards sharing an interrupt. The returned information includes the board number, the type of message, the subtype of the message, arguments for the port, timeslot, and other information contained in the message, and a character array that may contain any additional information. Query and version responses are received on a separate query queue and may be retrieved using the **xds_query_receive** function. Messages sent by the board may indicate a change in state of a port, an acknowledgement, or some sort of error condition. The message type and subtype can be used to decode this information

Message Sent
None

Returns
If a message is contained in the structure, a 0 will return. If no message was available on the receive queue, a 1 will be returned.

Comments
Message strings starting with ‘V’ or ‘Q’, or which have a ‘Q’ as the second character and do not have an ‘E’ or an ‘S’ for the first character are version request or query request responses and are returned on the query queue and accessed through the **xds_query_receive** function. All other messages are returned on the receive queue and accessed through this function.

On the BRI boards, there is an auxiliary mailbox. This mailbox is used to pass Layer 3 messages. The presence of valid data in this mailbox is indicated by a message in the mailbox that begins with “LC” or “LR”.

Example

xds_msg_receive(&rcv);	returns the message in the structure **rcv**

if the first message on the queue was the DID number detected message “SD051234” indicating that the DID address digits 1234 have been received on port 05, and the message was from board 2 the receive message structure would contain the following values:

rcv.board_number = 2;	board number 2
rcv.msg_type = 1;	a state change message
rcv.msg_sub = 3;	a DID detect message subtype
rcv.port_number = 5;	port 5 on the board
rcv.argument = 0;	no argument for this message subtype
rcv.msg_str[] = “1234”	the DID number detected
rcv.augRxLen	not valid data
rcv.augRxMesg	not valid data

xds_msg_send

int xds_msg_send(board_number, message);

unsigned char board_number;	a value indicating which board the command is for
char *message;	a pointer to a NULL terminated ASCII string which is the message to be sent

Applicable Boards
All XDS boards

Purpose
This function allows an application to send a message (command string) directly to an XDS board bypassing the library functions.

Message Sent
Determined by the application (user)

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The details of the board message sets can be found in the relevant manual for that board.

Example

xds_msg_send(1, “CD00”);	this will send the message “CD00” to board 1

xds_query_receive

int xds_query_receive(queryp);

struct rcvmsg query, *queryp {

unsigned char board_number;	a value indicating which board the command is for
char msg_type;	a value denoting the message type
char msg_sub;	a value denoting the message subtype
int port_number;	the port number, if any, involved in the particular message. if not port specific, this value should be (-1)
unsigned char argument;	a generalized argument field for the message whose meaning is specific to the particular message type, it will not have meaning for all message types
char msg_str[32];	a character array to pass any message specific values that do not fit in the other members of the structure
unsigned short augRxLen;	length of Layer 3 message in the auxiliary mailbox
unsigned char augRxMesg[260];	body of Layer 3 message in the auxiliary mailbox

}

Purpose
This function returns a structure with information about query or version request response messages sent from XDS boards. A single query queue is used for all XDS boards sharing an interrupt. The returned information includes the board number, the type of message, the subtype of the message, arguments for the port, timeslot, and other information contained in the message, and a character array that may contain any additional information. Query responses are used for diagnostic purposes. Version responses are used to obtain firmware version information. All query and version request responses are message type 16. The message subtype can be used to decode the message.

Message Sent
None

Returns
If a message is contained in the structure, a 0 will return. If no message was available on the query queue, a 1 will be returned.

Comments
Query response messages are hardware specific and require knowledge of the design of a particular board and the IC's used to implement the design. Version request information may be used to get the board ID and additional information about the board type, firmware version, and number of ports. However, as this requires sending messages in both directions, it is more efficient to use the **xds_id** function to get this information.

Example

xds_query_receive(&query);	returns the message in the structure **query**

if the first message on the query queue was the query clock mode response message "QC1" and the message was from board 2 the query message structure would contain the following values

query.board_number = 2;	board number 2
query.msg_type = 16;	a query response message
query.msg_sub = 4;	a clock mode query response message subtype
query.port_number = -1;	no port involved
query.argument = 0x31;	clock mode 1, encoded as ASCII
query.msg_str[] = 0	a NULL string
query.augRxLen	not valid data
query.augRxMesg	not valid data

xds_reset_all

int xds_reset_all(board_number);

unsigned char board_number;	a value indicating which board the command is for

Applicable Boards
All XDS boards

Purpose
This function can be used to put a board in a known, initialized state. All ports are released, all connections are broken, and all resources are freed. The CT bus is tri-stated with no output. This command does not change the clock mode of the board.

Message Sent
“RA”

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function should be used for all XDS boards when starting an application to put the boards in a known state.

Example

xds_reset_all(1);	resets the XDS board number 1

xds_set_encoding

int xds_set_encoding(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a character indicating the encoding mode, ‘A’ for A-Law, ‘M’ for Mu-law

Applicable Boards
All XDS boards

Purpose
This function is used to set the encoding mode for an XDS board. Two standards apply, A-Law which is used in Europe and Asia, and Mu-Law which is used in North America and Japan.

Message Sent
“SEm” where **m** is the mode.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid value for the encoding was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
There are two standards for the conversion between analog and digital and digital and analog signals used in telephony. These are A-Law, which is used in Europe and Asia, and Mu-Law, which is used in North America and Japan. These two standards are incompatible. Because of this, all boards in a system must use the same standard. If any outside digital lines such as T1, E1, or ISDN are used, the standard used must be compatible with that used by the digital line. Boards that only pass through digital signals such as the MVIP/SCSA Bridge Board do not require this function. However, any board, even if all digital that processes signals, such as with conferencing, must be set to the appropriate encoding standard. The default standard for XDS boards is Mu-Law.

Example

xds_set_encoding(1, ‘M’);	sets the encoding to Mu-Law board

xds_set_type

int xds_set_type(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char *types;	a pointer to a character array indicating port types

Applicable Boards
XDS boards with physical ports (interfaces)

Purpose
This function will set the behavior types of each port on the board. The allowed types of behavior will depend on the specific board types. As an example, setting a port to type 'P' or phone on a station board will allow that port to be rung. Setting the port type is important for the port to exhibit the correct behavior.

Message Sent
"ST(p * ports)" where **p** is the port type selected line boards and all non-H.110 boards.
"STL(p * ports)" where **p** is the mode – lower 16 ports (H.110).
"STH(p * ports)" where **p** is the mode – upper 16 ports (H.110).

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_ARG	an empty string of port types was passed to the function
ILL_TYPE	an invalid port type was selected for the board
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
As there are many port types for the different boards, you will need to consult the board reference manual to determine the valid types.

Example

xds_set_type(1, "NNUU");	sets ports 0 and 1 to "NT" and ports 3 and 4 to "Undefined" (BRI 4-port board)

xds_reset_dsp

int xds_reset_dsp(board_number);

unsigned char board_number;	a value indicating which board the command is for

Applicable Boards
XDS boards with reset-able DSPs

Purpose
This function should be used for maintenance on XDS boards that have reset able DSPs.

Message Sent
“RD”

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function should be used for maintenance on XDS boards that have reset-able DSPs.
Consult the hardware reference manual to check for this option.

Example

xds_reset_dsp(1);	resets the XDS board number 1

xds_num_port

int xds_num_port(board_number);

unsigned char board_number;	a value indicating which board the command is for

Applicable Boards
All XDS boards

Purpose
This function is used to query the number of ports (interfaces) on an XDS board

Message Sent
NONE

Returns
(-1) will be returned if there was an error querying the number of ports on the specified board. If this function is successful, it will return the number of ports that are present. It is possible that a board has 0 ports.

Comments
PCI-based Multi-chassis boards, such will return a (0) for the number of ports, unless the user assigns the number of ports, using the “SPstt” command.

Example

xds_num_port(1);	returns the number of ports for XDS board number 1

xds_set_config

xds_set_config(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a character indicating the mode of the configuration command, allowed values are ‘C’, ‘L’, ‘S’, and ‘B’

Applicable boards
All XDS ISA BRI boards and all H.100 and H.110 boards with the exception of the MC-3 board

Purpose
This function is used to clear, load, or save data in the EEAROM. This information consists of port type, protocol layer, directory numbers, and SPIDs. If this information is saved in the EEAROM, it is automatically loaded when the board reboots, eliminating the need for the application to download this configuration information.

Message Sent
“SMm” where **m** is the mode, valid modes are:

‘C’	- clear EEAROM data
‘L’	- load EEAROM data into RAM memory
‘S’	- save configuration information in RAM in the EEAROM
‘B’	- transfer configuration buffer to active memory

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	invalid configuration mode
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to save the current configuration information held in RAM in the on board EEAROM. This information will then be loaded into RAM every time the board reboots. This saves the application from having to download the configuration information every time the board reboots. It is only necessary to save information when the configuration of the board changes. If a number of changes are made at a time, it is only necessary to save the information after the last change has been made.

The information saved consists of the port types, the support layer selected, the clock mode, and on the BRI interface boards, information such as the directory numbers and SPIDs associated with each BRI interface.

Example

xds_set_config(16, ‘S’);	this will cause board 16 to save configuration information

xds_queue_message

xds_queue_message (board_number, message);

unsigned char board_number;	a value indicating which board the command is for
char *message;	a character string indicating the message string to be queued

Applicable boards
All XDS boards

Purpose
This function is used to place an XDS message string on the message receive queue for a requested board.

Message Sent
None

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Example

xds_queue_message (16, “SH00”);	this will place the “SH00” message on the receive queue

H.100 Generic Board Functions

This page was intentionally left blank.

xds_set_ctbus_rate

xds_set_ctbus_rate(board_number, rate1, rate2, rate3, rate4);

unsigned char board_number;	a value indicating which board the command is for
char rate1;	indicates the rate for streams 0-3
char rate2;	indicates the rate for streams 4-7
char rate3;	indicates the rate for streams 8-11
char rate4;	indicates the rate for streams 12-15

Applicable boards
All Infinity Series H.100 boards

Purpose
This function is used to control the bit rate for the first 16 streams on the H.100 bus.

Message Sent
“SBabcd” where **a**, **b**, **c**, and **d** are the bit rate arguments

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid bus rate was selected
WRONG_BOARD	a non-H.100 board was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The H.100 specification allows the first 16 streams of the H.100 bus to operate at 2, 4 or 8 MHz. for compatibility with legacy CT busses such as the MVIP and H.100 bus. This function controls the bit rate on these streams. For compatibility with the H.100 bus operating at 2 MHz., the rate arguments should all be 0x0. Normally, the stream rate is set during initialization.

Example

xds_set_ctbus_rate(16, 0, 0, 0, 0);	this sets streams 0-15 to operate at 2 MHz.
xds_set_ctbus_rate(16, 1, 1, 1, 1);	this sets streams 0-15 to operate at 4 MHz.
xds_set_ctbus_rate(16, 2, 2, 2, 2);	this sets streams 0-15 to operate at 8 MHz.

xds_set_ct_clock_mode

xds_set_ct_clock_mode(board_number, mode, submode, arg1, arg2, arg3);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a value between 0 and 5 selecting the clock mode
char submode;	a value between 0 and 7 selecting the clock submode
char arg1;	a value between 0 and 3 specifying a clock mode argument
char arg2;	a value between 0 and 5 specifying a clock mode argument
char arg3;	a value between 0 and 3 specifying a clock mode argument

Applicable boards
All Infinity Series H.100/H.110 Boards

Purpose
This function is used to set the clock mode for XDS H.100 boards. Because the clocks for the MC3 bus and H.100/H.110 bus are related, this function may affect clocks on both busses.

Message Sent
"Scmsabb(c)" where **m** is the clock mode, **s** is the submode, **a** is arg1, **bb** is arg2, and **c** is agr3.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid clock mode was selected
WRONG_BOARD	a non-H.100/H.110 board was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
In a system of PCs tied together using the MC3 boards, there must be only one source of the master clock. This clock may be generated by any board in the system and may be created by a local oscillator on that board or derived from an outside source such as a T1 span. All other boards in the system must be slaved to this clock. The MC3 Board in the same chassis with the master clock must derive the MC3 bus clock from the H.100/H.110 bus and generate the clocks for the MC3 bus. Other MC3 Boards in the system must derive their clocks from the MC3 bus and supply the clocks to the H.100/H.110 bus in their chassis. It is possible for a MC3 Board to supply all the clocks to both buses from an on board oscillator.

Because all boards must be set to compatible clock modes, this function must be used with care. A more detailed discussion of the clocks available to each board may be found in any of the Infinity Series board technical manuals.

Example

xds_set_ct_clock_mode(16, 2, 3, 0, 2);	this will set board 16 to derive the clock from the Ring 0 of the MC3 bus and provide the clock to the H.100/H.110 bus.

xds_set_h100_termination

xds_set_h100_termination(board_number, h_bus, m_bus);

unsigned char board_number;	a value indicating which board the command is for
char h_bus;	a character indicating whether the H.100 termination should be enabled 'E', or disabled 'D'
char m_bus;	a character indicating whether the MVIP bus termination should be enabled 'E', or disabled 'D'

Applicable boards
All Infinity Series H.100 Boards

Purpose
This function is used to control the termination for the H.100 bus and the MVIP bus.

Message Sent
"STab" where **a** is the H.100 bus termination mode and **b** is the MVIP bus termination mode.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid termination type was selected
WRONG_BOARD	a non-H.100 board was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The H.100 and MVIP-90 specifications require that the boards on the ends of these busses provide termination while boards in the middle of the busses do not. This function controls whether the H.100 board provides that termination. This function has no affect when using the board in SCSA Legacy mode.

Example

xds_set_h100_termination(16, 'E', 'D');	this enables the H.100 termination and disables the MVIP bus termination on board 16

xds_pci_resources

int xds_pci_resources(board_number, info);

unsigned char board_number;	a value indicating which board the command is for
struct xds_id*info;	a structure to contain the return information
{	
char version[5];	an array containing the firmware version number
char id[5];	an array containing the board ID string
UCHAR num_ports;	the number of ports on the board
UCHAR board_number;	the board number
ULONG pci_bus_number;	the PCI bus number
ULONG pci_slot_number;	the PCI slot number
}	

Applicable Boards
All Infinity Series H.100/H.110 Boards

Purpose
This function will return the PCI bus and slot (device) number for the board. This information is returned in a structure of the type **xds_id**.

Message Sent
None.

Returns
This function will return a 0 if successful. If it is not successful, it will return a 1.

Example

xds_pci_resources(16, &info);	queries the PCI bus and device number for board 16

xds_board_reset

int xds_board_reset(board_number);

unsigned char board_number;	a value indicating which board the command is for

Applicable Boards
All Infinity Series H.100/H.110 Boards

Purpose
This function will reset the entire board without having to have the user shut down the system.

Message Sent
“IF”, then the hardware reset

Returns
This function will return a 0 if successful. If it is not successful, it will return a 1.

Comments
This function could be useful if a board becomes congested. After the board is active again, the user will need to resend a “IN” to re-enable board interrupts (if they choose).

Example

xds_board_reset(16);	resets board number 16

xds_check_ctbus_rate

xds_check_ctbus_rate(board_number);

unsigned char board_number;	a value indicating which board the command is for

Applicable boards
All Infinity Series H.100 boards

Purpose
This function is used to check the bit-rate for the first 16 streams on the H.100 bus.

Message Sent
None.

Returns

'M'	MVIP 2 MHz Legacy mode
'S'	SCSA 4 MHz Legacy mode
'H'	H.100 8 MHz mode

or if there was a DPRAM read error:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The H.100 specification allows the first 16 streams of the H.100 bus to operate at 2, 4 or 8 MHz. for compatibility with legacy CT busses such as the MVIP and H.100 bus. This function controls the bit rate on these streams. For compatibility with the H.100 bus operating at 2 MHz., the rate arguments should all be 0x0. Normally, the stream rate is set during initialization.

Example

xds_check_ctbus_rate(16);	this will check the mode on board 16.

This page was intentionally left blank.

Basic Rate ISDN Common Functions

This page was intentionally left blank.

xds_bri_ctbus_cptones

int xds_bri_ctbus_cptones(board_number, tone, stream, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int tone;	a value between 0x20 and 0x25 or 0x40 and 0x45, depending on type of board, indicating which call progress tone to play
int stream;	a value between 0x0 and 0xF (ISA), or 0x0 and 0x1F (H.100 & H.110) indicating the stream the tone should be output on
int timeslot;	a value between 0x0 and 0x3F (ISA), or 0x0 and 0x7F (H.100 & H.110) indicating the timeslot on that stream

Applicable boards
All XDS BRI boards except for the MVIP (ISA) version.

Purpose
This function transmits a call progress tone to the CT bus. Certain Layer 3 commands may indicate that a call progress tone should be sent to the listener. This command allows this if the board to which the listener is connected does not have call progress facilities.

Message Sent
"CSxxsstt" where **xx** is the tone, **ss** is the stream, and **tt** is the timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an out of range call progress tone was selected
WRONG_BOARD	an MVIP BRI board was selected (not supported)
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The purpose of this command is to allow for the playing of a call progress tone onto the CT bus when a received Layer 3 message indicates that this is appropriate with a signal information element. As not all MVIP ISA boards are capable of generating call progress tones, provisions were made for this facility on the other forms of the BRI board. The stream and timeslot value are the actual stream and timeslot on the CT bus.

The six tones produced by this function are selected with the tone argument. The tones produced are:
SCSA ISA and H.100 PCI board-
0x20 - dial tone
0x21 - reorder/ congestion tone
0x22 - busy tone
0x23 - audible ringback
0x24 - digital milliWatt, a 1004 Hz. steady tone
0x25 - silence

H.110 cPCI board-
0x40 - dial tone
0x41 - reorder/ congestion tone
0x42 - busy tone
0x43 - audible ringback
0x44 - digital milliWatt, a 1004 Hz. steady tone
0x45 - silence

The first four tones are as defined in RS-464.

Example

xds_bri_ctbus_cptones(16, 0x21, 2, 3);	this will play "reorder tone" on the CTbus stream 2, timeslot 3 timeslot of (H.100) board 16

xds_bri_bchannel_cptones

int xds_bri_bchannel_cptones(board_number, tone);

unsigned char board_number;	a value indicating which board the command is for
int tone;	a value between 0x20 and 0x25 or 0x40 and 0x45, depending on type of board, indicating which call progress tone to play

Applicable boards
All XDS BRI boards

Purpose
This function gives a call progress tone to a specified B-channel.

Message Sent
"CPxxy" where **xx** is the B-channel and **y** is the tone.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
ILL_ARG	an out of range call progress tone was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The purpose of this command is to allow for the playing of a call progress tone onto the B-channel when a received Layer 3 message indicates that this is appropriate with a signal information element.

The six tones available to this function are selected with the tone argument. They are:

0 - dial tone
1 - reorder/ congestion tone
2 - busy tone
3 - audible ringback
4 - digital milliWatt, a 1004 Hz. steady tone
5 - silence

Example

xds_bri_bchannel_cptones(16, 2, 1);	this will play "reorder tone" on B-channel 2, board 16

xds_bri_deactivate_layer1

xds_bri_deactivate_layer1(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface

Applicable boards
All XDS BRI boards, except the ISA S/T boards

Purpose
This function is used to deactivate Layer 1 on an NT or LT type port.

Message Sent
“RIpp” where **pp** is the port number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will cause Layer 1 to be deactivated on an S/T port set to type NT or a U-Interface port set to type LT. With Layer 1 deactivated, no communication including the B-channels is possible until Layer 1 is reactivated. This function is present in S/T boards with firmware version 0.7 or later or EuroISDN firmware and U-interface boards with firmware version 0.0h or later.

Example

xds_bri_deactivate_layer1(16, 0x4);	this will deactivate port 4 on board 16

xds_bri_loopback

int xds_bri_loopback(board_number, port, mode);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int mode;	a value between 0 and 3 indicating the loopback mode

Applicable boards
All XDS BRI boards

Purpose
This function is used to set the loopback mode on the BRI interface specified with the port argument. Setting the loopback mode can be used for diagnostic purposes.

Message Sent
“XLxxm” where **xx** is the BRI interface number and l is the loopback mode

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
ILL_MODE	the mode specified is outside the range 0x0 to 0x3
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to place a BRI interface into loopback at the S/T interface. The possible modes are:
0 - no loopback
1 - loopback on channel B1
2 - loopback on channel B2
3 - loopback on both channels

This command should only be used for diagnostic purposes. The interface will not operate properly if left in a loopback condition.

Example
xds_bri_loopback(1, 2, 3);   this will loopback both channels on BRI interface 2 of board 1.

xds_bri_power_feed

xds_bri_power_feed(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char *mode;	a pointer to a character array indicating the power feed mode for each BRI interface.

Applicable boards
XDS H.100 & H.110 S/T and 4 port BRI boards

Purpose
This function is used to control the power feed circuit for each NT port.

Message Sent
"SF(m * 8)" where **m** is the power feed mode information – 4 port.
"SF(m * 16)" where **m** is the power feed mode information, H.100.
"SFL(m * 16)" where **m** is the power feed mode information, lower bank – H.110.
"SFH(m * 16)" where **m** is the power feed mode information, upper bank – H.110.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	invalid power feed mode selected
ILL_ARG	the power feed mode parameter was NULL
WRONG_BOARD	an non-BRI board was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call
SUCCESS	the function returned successfully

Comments
When set to type NT, an interface may supply power to the attached terminal. This function controls the power feed circuit. Ports that are set to type TE should have the power feed mode set to ‘F’.

The mode parameter is a string of characters that is used to control the battery feed mode for each port. The valid parameters for the mode string are a ‘N’, ‘F’, or ‘*’ for each port. The ‘N’ mode will turn the power feed on for a given port, ‘F’ will turn the power feed off for a given port, and ‘*’ will leave the mode unchanged. A value must be entered for each of the ports.

On the 4 port BRI board, the first four characters control the PS1 power for the ports and the last four characters control the PS2 power for the ports. On the H.100 BRI S/T board, the sixteen characters control the PS1 power for the ports. On the H.100 BRI S/T board, the sixteen characters control the PS1 power for the ports.

Example

4 port-	
xds_bri_power_feed(16, “NNNNFFFF”);	this will turn the battery feed on for all for of the PS1 ports and turn the battery feed off for all four of the PS2 ports on board 16
H.100-	
xds_bri_power_feed(17, “NNNNNN**FFFFFF**”);	this will turn the battery feed on for ports 0 - 5, leave ports 6 and 7 unchanged, turn battery feed off on ports 8 – 13, and leave ports 14 and 15 unchanged on PS1 of board 17
H.110-	
xds_bri_power_feed(5, “NNNNNNNNNNNNNNNN”);	this will turn the power feed on for the lower bank of 16 ports on board 5
xds_bri_power_feed(5, “FFFFFFFFFFFFFFFF”);	this will turn the power feed off for the upper bank of 16 ports on board 5

xds_bri_reactivate_layer1

xds_bri_reactivate_layer1(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface

Applicable boards
All XDS BRI boards, except the ISA S/T boards

Purpose
This function is used to reactivate Layer 1 on an NT or LT type port.

Message Sent
"RLpp" where **pp** is the port number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will cause the board to attempt to reactivate Layer 1 on a port. If Layer 1 reactivation is successful, Layers 2 and 3 will be reestablished. This function is present in S/T boards with firmware version 0.7 or later or EuroISDN firmware and U-interface boards with firmware version 0.0h or later.

Example

xds_bri_reactivate_layer1(16, 0x4);	this will reactivate layer 1 on port 4 of board 16

xds_bri_reset_bchannel

xds_bri_reset_bchannel(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the Layer 3 B-channel interface

Applicable boards
All XDS BRI boards, except the ISA S/T boards

Purpose
This function is used to reset a B-channel at Layer 3. This involves clearing the call state and call reference for the selected B channel.

Message Sent
“RBxx” where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function resets Layer 3 on the selected B-channel. The call state for that B-channel is set to idle (U0), and the associated call reference is released. This function is present in S/T boards with firmware version 0.7 or later or EuroISDN firmware and U-interface boards with firmware version 0.0h or later. This command does not disable connections to the SCbus.

Example

xds_bri_reset_bchannel(16, 0x4);	this will reset B-channel 4 on board 16

xds_bri_test

int xds_bri_test(board_number, port, mode);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int mode;	a value between 0 and 2 indicating the test mode

Applicable boards
All XDS BRI boards

Purpose
This function is used to place the BRI interface specified with the port argument into a test mode for diagnostic purposes.

Message Sent
"XTxxm" where **xx** is the BRI interface number and **m** is the test mode.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
ILL_MODE	the test mode is invalid
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to place a BRI interface into one of two test modes. In these modes, the interface will output a square wave at either 2 kHz (mode 1) or 96 kHz (mode 2). Mode 0 clears the test signal. This command should only be issued for test purposes. The interface will not operate when in the test mode.

Example

xds_bri_test(1, 2, 2);	this will place BRI interface 2 of board 1 in test mode 2.

xds_bru_eoc

int xds_bru_eoc(board_number, port, mode)

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int mode;	a value between 0 and 6 indicating the EOC command

Applicable boards
XDS U-Interface BRI Boards

Purpose
This command is used to send "Embedded Operation Channel" messages from a "Line Termination" (LT) port for maintenance functions.

Message Sent
"XExxm" where **xx** is the BRI Interface number and **m** is the EOC command mode

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
ILL_MODE	the mode is invalid
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Embedded Operations Channel is used to send commands from an Line Termination to a Network Termination to put the NT in loopback or other test modes. The EOC command modes are:
0 - RTN, return to normal
1 - LB1, loopback of channel B1
2 - LB2, loopback of channel B2
3 - LBBD, loopback of both B channels and the D channel
4 - RRC, request corrupt CRC
5 - NCC, notify of corrupt CRC
6 - H, Hold

Example

xds_bru_eoc(1, 2, 3)	will send the LBBD EOC command on port 2 of board 1

xds_bru_mon8

int xds_bru_mon8(board_number, port, mode)

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int mode;	a value between 0 and 7 indicating the MON-8 command

Applicable boards
XDS U-Interface BRI Boards

Purpose
This command is used to send local command messages to the U-Interface hardware for maintenance functions.

Message Sent
"XLxxm" where **xx** is the BRI Interface number and **m** is the "MON-8" command mode

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
ILL_MODE	the mode is invalid
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The "MON-8" commands are used to place the U-Interface hardware in maintenance or test modes. The "MON-8" command modes are:
0 - PACE, Partial Activation Control External
1 - PACA, Partial Activation Control Automatic
2 - CCRC, Corrupt CRC (LT only)
3 - LBBD, loopback of both B channels and the D channel (NT only)
4 - NORM, return to normal
5 - RBEN, read Near-End Block Error Counter
6 - RBEF, Read Far-End Block Error Counter
7 - SFB, set FEBE bit to 0

Example

xds_bru_mon8(1, 2, 3)	will send the "LBBD MON-8" command to port 2 of board 1

xds_reset_port

xds_reset_port(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface

Applicable boards
All XDS BRI boards

Purpose
This function is used to reset a port. This involves a reset of the Layer 1 and Layer 2 protocols on the port.

Message Sent
"RPpp" where **pp** is the port number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function can be used to force a reset of the Layer 1 and Layer 2 protocols on a port. All data links for that port will be disconnected and the terminal endpoint identifiers (TEIs) associated with the port will be removed. After reestablishing Layer 1, Layer 2 procedures will be followed to assign new TEIs. This command does not disable connections to the CT bus.
The board will respond with a Port Reset Acknowledge message.

Example

xds_reset_port(1, 0x4);	this will reset port 4 on board 1

xds_set_bri_dn

xds_set_bri_dn(board_number, b_channel, dn, spid);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	the number of the b-channel that indicates the BRI interface
char *dn;	a pointer to a character array containing a seven digit ASCII string with the directory number
char *spid;	a pointer to a character array containing an ASCII string with the SPID (TE interfaces only) the SPID may be up to 14 digits

Applicable boards
All XDS BRI boards

Purpose
This function is used to set the default DN or directory number associated with a particular B-channel. If the interface is defined as terminal equipment or TE on an S/T Interface Board or as a Network Termination or NT on a U-Interface Board, this function is also used to set the Service Profile ID or SPID. This function is only relevant when layer 3 support is selected.

Message Sent
"SDddddddd" where **ddddddd** is the directory number for the B-channel, or
"SDddddddd/sssssssssssss" for a TE interface where **sssssssssss** is the SPID.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	invalid length for the directory number
ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
If layer 3 support is chosen using the **xds_set_layer**, it is necessary to define a default directory number for each B-channel. On a port defined as a network termination or NT, the number is used as the called party number in SETUP messages. If the port is defined as a TE, the number is the calling party in SETUP messages. It is possible for both B-channels on an interface to have the same directory number.

For a port defined as a TE, it is also necessary to define one or two SPID's. If only one SPID is required, it is defined for the even B-channel of that port. No SPID is defined for the odd B-channel. If two SPID's are required, one SPID is defined for each B-channel. The data-link associated with the SPID set for the even B-channel will be used for speech and audio calls while the data-link associated with the SPID set for the odd B-channel will be used for data calls. Note that this does not restrict speech calls to the even B-channel and data calls to the odd B-channel. The directory number and SPID can be save in EEAROM using the **xds_set_config** function. If this is done, this and other information will automatically be restored on a reboot of the board. Therefore, this function will only be needed if the configuration changes.

Example

xds_set_bri_dn(1, 0x4, "5551000", '');	this will set the default DN for B-channel 4 to 555-1000
xds_set_bri_dn(1, 0x6, "5552000", "608555200001")	this will set the default DN for B-channel 6 to 555-2000 and the SPID to 608555200001.

xds_set_layer

xds_set_layer(board_number, layer);

unsigned char board_number;	a value indicating which board the command is for
char *layer;	a pointer to a character array indicating the protocol layer for each BRI interface, the array consists of a character for each interface on the board. Valid characters are ‘2’, ‘3’, ‘A’, ‘D’, ‘E’, or ‘N’.

Applicable boards
All XDS BRI boards

Purpose
This function is used to set the protocol layer for each of the BRI interfaces. The board can provide support at layer 2, layer 3, AT&T Custom, DMS-100, CACH EKTS, or National ISDN 1.

Message Sent
“SL(p*number of ports)” (ISA / H.100 boards) where p is the protocol layer
“SLL(p*number of ports)” (H.110 boards) where p is the protocol layer, lower bank of 16
“SLH(p*number of ports)” (H.110 boards) where p is the protocol layer, upper bank of 16

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	invalid layer type
WRONG_BOARD	a non-BRI board was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The board will provide support at either layer 2 or layer 3. If layer 2 support is chosen, the board will handle all layer 2 functions, but will require the application to compose and interpret the raw Q.931 messages for call control. These messages are sent using the **xds_l3_message()** function**.** If layer 3 support is chosen, call control communications between the driver and the board is accomplished through ‘D’ messages. This is a simpler application interface and does not require the program to have as detailed a knowledge of the Q.931 message structure.

Layer types are as follows:

2 - basic Layer 2 support
3 - basic Layer 3 support
A - AT&T Custom support
D - DMS-100 or 5E National ISDN-1
E- CACH EKTS support
N - National ISDN-1 (Siemens EWSD switch)

The default support is layer 2. The protocol layer supported can be saved in EEAROM using the **xds_set_config** function. If this is done, the support layer and other information will automatically be restored on a board reboot. Therefore, this function will only be needed if the configuration changes.

Example

xds_set_layer(1, “333333222222”);	this will set board 1 to support layer 3 on the first six interfaces and layer 2 on the last six.

xds_tei_assign

int xds_tei_assign(board_number,port);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface

Applicable boards
All XDS BRI boards

Purpose
This function is used to request the assignment of a Terminal Endpoint Identifier (TEI) to the BRI interface specified with the port argument. The port must be defined as type TE, terminal equipment. At least one TEI is needed for sending Layer 3 call control messages.

Message Sent
"TAxx" where **xx** is the BRI interface number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to request the TEI portion of the DLCI (Data Link Connection Identifier) address. This address is used to specify which data link a Layer 3 message is for in the LAPD protocol. Only TE (terminal equipment) on an S/T Board or NT (network terminatiion) on a U-Interface Board ports can request a TEI. The TEI will be granted by the NT (network termination) or LT (line termination) end of the ISDN circuit. The XDS MVIP BRI Boards can handle the assigment of up to seven TEIs for each BRI interface.
When a TEI has been assigned, the TEI will be reported in a TEI assignment message of type 9 sub_type 4.

Example

xds_tei_assign(1, 2);	this will request a TEI on BRI interface 2 of board 1.

xds_tei_check

int xds_tei_check(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface

Applicable boards
All XDS BRI boards

Purpose
This function is used to request that a Terminal Endpoint Identifier (TEI) check procedure be carried out on the BRI interface specified with the port argument. The port must be defined as type NT, network termination for an S/T Board or as type LT for a U-Interface Board. A TEI check procedure is used to verify that no conflict exists in TEI assignment.

Message Sent
"TCxx" where **xx** is the BRI interface number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to request that a TEI check procedure be carried out. This procedure is used to verify that no TEI is assigned to more than one TE terminal equipment, and that a TE exists for each TEI assigned. If there is a TEI assignment error, this procedure will detect and correct it. Only an NT, network termination, can initiate a TEI check procedure.

Example

xds_tei_check(1, 2);	this will request a TEI check on BRI interface 2 of board 1.

xds_tei_cnct

int xds_tei_cnct(board_number, port, tei);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int tei;	a value between 0x0 and 0x7E indicating the TEI of the data link to be established, a value of 0x7F indicates the packet data link

Applicable boards
All XDS BRI boards

Purpose
This function is used to request the establishment of the DLCI specified by a SAPI of 0 and the Terminal Endpoint Identifier (TEI) specified or the packet data link on the BRI interface specified with the port argument.

Message Sent
"TExxtt" where **xx** is the BRI interface number and **tt** is the TEI number.
"TExxP" where **xx** is the BRI interface number and P indicates the packet data link.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_ARG	the TEI specified is outside the range 0x0 to 0x7F
ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to request that the data link specified by DLCI (Data Link Connection Identifier) address of [0, TEI] be established. If the TEI value is 0x7F, the packet data link with a SAPI of 16 is specified. The data link must be in the TEI assigned state. If successful, the data link will be put into the multi-frame established state.

Example

xds_tei_cnct(1, 2, 0x40);	this will request establishment for a TEI of 64 (0x40) on BRI interface 2 of board 1.

xds_tei_disc

int xds_tei_disc(board_number, port, tei);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int tei;	a value between 0x0 and 0x7E indicating the TEI of the data link to be disconnected, a value of 0x7F indicates the packet data link

Applicable boards
All XDS BRI boards

Purpose
This function is used to request the disconnection of the DLCI specified by a SAPI of 0 and the Terminal Endpoint Identifier (TEI) specified or the packet data link on the BRI interface specified with the port argument.

Message Sent
"TDxxtt" where **xx** is the BRI interface number and **tt** is the TEI number.
"TDxxP" where **xx** is the BRI interface number and P indicates the packet data link.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the TEI specified is outside the range 0x0 to 0x7F
ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to request that the data link specified by DLCI (Data Link Connection Identifier) address of [0, TEI] be disconnected. If the TEI argument is 0x7F, the packet data link with a SAPI of 16 is specified. Disconnection does not cause the removal of the TEI, but does place it in the TEI assigned state. The link may be reconnected by an attempt to send a message or a call to **xds_tei_cnct**.

Example

xds_tei_disc(1, 2, 0x40);	this will request a disconnection for a TEI of 64 (0x40) on BRI interface 2 of board 1.

xds_tei_fixed

int xds_tei_fixed(board_number, port, tei);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int tei;	a value between 0x0 and 0x3F indicating the fixed TEI

Applicable boards
All XDS BRI boards

Purpose
This function is used to specify a fixed Terminal Endpoint Identifier (TEI) to be used on the BRI interface specified with the port argument.

Message Sent
"TFxxtt" where **xx** is the BRI interface number and **tt** is the TEI number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_ARG	the TEI specified is outside the range 0x0 to 0x3F
ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to specify a fixed TEI (Terminal Endpoint Identifier) so that it will be available for sending or receiving Layer 3 messages. Fixed TEI are in the range 0x0 to 0x3F. Each port may have up to seven TEI's assigned. This number includes both fixed and dynamic TEI's. Normally, it is only necessary to specify a fixed TEI for a TE (terminal equipment) port. However, this command may also be used to reserve a TEI on an NT (network termination) port if there is a possibility that the assignment of dynamic TEIs may take up all seven possible TEIs. Dynamic TEI assignment using the **xds_tei_assign** command is to be prefered where it is an option.

Example

xds_tei_fixed(1, 2, 0x20);	this will specify a TEI of 32 (0x20) on BRI interface 2 of board 1.

xds_tei_packet

int xds_tei_packet(board_number, port, tei);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int tei;	a value between 0x0 and 0x7E indicating the packet TEI

Applicable boards
All XDS BRI boards

Purpose
This function is used to specify a packet Terminal Endpoint Identifier (TEI) to be used on the BRI interface specified with the port argument.

Message Sent
“TPxxtt” where **xx** is the BRI interface number and **tt** is the TEI number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the TEI specified is outside the range 0x0 to 0x7F
ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to specify a packet TEI (Terminal Endpoint Identifier) so that it will be available for sending or receiving Layer 3 messages with a SAPI of 16. The Packet TEI are in the range 0x0 to 0x7E. Each port supports a single packet data link that may be used for sending X.31 messages on the D channel. This is in addition to the seven fixed and dynamic TEI’s links available for a SAPI of 0. The default packet TEI is 0.

Example

xds_tei_packet(1, 2, 0x20);	this will specify a TEI of 32 (0x20) on BRI interface 2 of board 1.

xds_tei_query

int xds_tei_query(board_number, port, teiList);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int teiList[8];	an array for the return of the TEIs assigned to the port

Applicable boards
All XDS BRI boards

Purpose
This function is used to request a list of the TEIs assigned to the on the BRI interface specified with the port argument.

Message Sent
“TQxx” where **xx** is the BRI interface number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
WRONG_QUERY	the wrong query response was received
WRONG_BOARD	a query response was received from the wrong board
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to request a list of TEIs (Terminal Endpoint Identifiers) that have been assigned on a BRI interface. This list will consists of eight TEIs, with the last TEI being that of the packet data link. Unassigned TEIs will have a value of 0xFF. Both dynamic and fixed TEIs will be reported. The broadcast TEI of 0x7F will not be included in the list, though it is available. The response message from the board will take the form “TQxxxxxxxxxxxxxxxx” where each pair of xx’s is a hexadecimal number representing a TEI.

Example

xds_tei_query(1, 2, &list);	this will request a TEI list to be placed in the array & list on BRI interface 2 of board 1.

xds_tei_remove

int xds_tei_remove(board_number, port, tei);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int tei;	a value between 0x0 and 0x7F indicating the TEI to be removed, 0x7F will cause all TEIs to be removed

Applicable boards
All XDS BRI boards

Purpose
This function is used to request the removal of Terminal Endpoint Identifier (TEI) specified on the BRI interface specified with the port argument.

Message Sent
“TRxxtt” where **xx** is the BRI interface number and **tt** is the TEI number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the TEI specified is outside the range 0x0 to 0x7F
ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to request the removal of the TEI specified. This will cause the DLCI (Data Link Connection Identifier) at [0, TEI] to be eliminated. A TEI value of 0x7F will cause all TEIs in the range 0x0 to 0x7E to be removed. This command is only valid for NT (network termination) ports on S/T BRI Boards and LT (line termination) ports on U-Interface BRI Boards. The removal will be reported with a message of type 9 msg_sub 5.

Example

xds_tei_remove(1, 2, 0x40);	this will request removal of TEI 64 (0x40) on BRI interface 2 of board 1.

xds_tei_verify

int xds_tei_verify(board_number, port, tei);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface
int tei;	a value between 0x0 and 0x7F indicating the TEI to be verified

Applicable boards
All XDS BRI boards

Purpose
This function is used to request the verification of the Terminal Endpoint Identifier (TEI) specified on the BRI interface specified with the port argument.

Message Sent
"TVxxtt" where **xx** is the BRI interface number and **tt** is the TEI number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the TEI specified is outside the range 0x0 to 0x7F
ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to request the verification of a TEI (Terminal Endpoint Identifier). This command is only valid on TE (terminal equipment) ports. The associated NT (network termination) or LT (line termination) will carry out the verification and make corrections if necessary by removing the TEI.

Example

xds_tei_verify(1, 2, 0x40);	this will request the verification of TEI 64 (0x40) on BRI interface 2 of board 1.

xds_l3_restart

xds_l3_restart(board_number, b_channel, channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	the number of the b-channel that indicates the BRI interface
int channel;	the number of the specific b-channel to be restarted

Applicable boards
All XDS BRI boards

Purpose
This function is used to restart a B-channel.

Message Sent
"DZxx" restart both B-channels, where **xx** is either of the B-channels
or
"DZxxC" restart a specific B-channel, where **xx** is the B-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the channel chosen was not 0 or 1
ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Restart Indicator is used in RESTART and RESTART ACKnowledge messages to indicate which B-channels are involved. The restart procedure lets either one or both of the B-channels to be returned to the null state.

Examples

xds_l3_restart(1, 0x4, 0);	this will restart both B-channels 4 on board 1
xds_l3_restart(1, 0x4, 1);	this will restart only B-channel 4 on board 1

xds_set_bri_clock

xds_set_bri_clock(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the BRI interface

Applicable boards
XDS SCSA S/T BRI and XDS SCSA U-Interface BRI Boards

Purpose
This function is used to select a BRI port as the master clock source for the system. This port must be configured as a TE (Terminal Equipment) interface for an S/T interface board and as an N/T for a U-interface board.

Message Sent
"SC2p" where **p** is the port number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-SCSA BRI board was chosen
ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The SCSA S/T BRI board has twelve Basic Rate ISDN interfaces while the SCSA U-Interface BRI board has four. This function can be used to select one of these interfaces to act as the source of the master clock for the system. This port must be configured as terminal equipment (TE) for a S/T board or as an N/T for a U-interface board and be connected to a Network Termination or Line Termination respectively, either a central office line or a line from a PBX or other switch intended to interface to terminal equipment. When this command is issued, the board will derive its clock from the BRI interface and supply the clocks to the SCbus. It is important that no other board in the system be supplying clocks to the SCbus when this function is called, as only one board can serve as master clock.

Example

xds_set_bri_clock(1, 0x4);	this will select port 4 on board 1 as the source of the system clock

Basic Rate ISDN NI-1
Layer 3 ‘D’ Functions

This page was intentionally left blank.

xds_l3_alert

xds_l3_alert(board_number, b_channel, progress, signal);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the progress indicator if any
char signal;	a character indicating the signal if any

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 ALERTing message for the call currently associated with the specified B-Channel. An ALERTing message is used to notify the caller that the called party is being informed of an incoming call.

Message Sent
"DAxxps" where **xx** is the B-channel number, **p** is the progress indicator, and **s** is the signal argument.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
ILL_SIGNAL	invalid signal
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an ALERTing message to be sent for the call associated with the specified B-channel. If the port is configured as an NT (network termination), progress indication and signal information elements can be sent as part of the message.
For NT ports a progress indicator is part of the ALERTing message. Valid values for this are:

I	Inband information or appropriate pattern now available
N	No progress indicator

A value of ‘I’ should be used for speech or 3.1 kHz audiocalls, while ‘N’ should be used for data calls. The signal element for NT ports should be set to ‘R’ for ringback.
For TE ports, the progress and signal arguments should be set to 0x0.

Example

xds_l3_alert(1, 0x4, ‘I’, ‘R’);	this will send an ALERTing message for B-channel 4 on board 1. The progress indicator is set for inband signal, the signal is ringback.

xds_l3_connect

xds_l3_connect(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 CONNect message for the call currently associated with the specified B-Channel. A CONNect message is used to notify the caller that the called party has answered an incoming call.

Message Sent
"DCxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a CONNect message to be sent for the call associated with the specified B-channel. This should be done when the called party answers.

Example

xds_l3_connect(1, 0x4);	this will send a CONNect message for B-channel 4 on board 1.

xds_l3_disconnect

xds_l3_disconnect(board_number, b_channel, cause, signal);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the disconnect
char signal;	a character indicating the signal if any

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 DISConnect message for the call currently associated with the specified B-Channel. A DISConnect message is used to notify either the network or the user that a disconnect sequence has been initiated.

Message Sent
"DDxxccs" where **xx** is the B-channel number, **cc** is the cause code, and **s** is the signal.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_SIGNAL	invalid signal value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This command is used to cause a DISConnect message to be sent for the call associated with the specified B-channel. This should be done when initiating a disconnect sequence.
The cause code is a value used to indicate the reason for a particular message, in this case, why a disconnect is occurring. A cause value of 0x10 indicates normal clearing. Other values may indicate error conditions. For a list of cause codes and there meanings, see the XDS Layer 3 Protocol Software Reference Manual, Q.850 or the relevant Bellcore documentation.
If the port is configured as an NT (network termination), a signal element can be included in the DISConnect message. Valid values for the signal element are:

R	audible ringing tone on
N	network congestion/reorder tone on
B	busy tone on
C	confirmation tone
F	tones off

Example

xds_l3_disconnect(1, 0x4, 0x10, 'F');	this will send a DISConnect message for B-channel 4 on board 1, with a cause of 0x10 (normal clearing), and tones off.

xds_l3_feature_ind

xds_l3_feature_ind(board_number, b_channel, feature, status);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int feature;	a value between 0x0 and 0x3F indicating the feature indicator
char status;	a character indicating the status of the feature indicator

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message to activate a feature indicator on an ISDN terminal or station set. Feature indicators are used to inform the user of the status of a particular feature.

Message Sent
“DFxxffa” where **xx** is the B-channel number, **ff** is the feature indicator code, and **a** is the feature status.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_FEATURE	invalid feature value
ILL_ARG	invalid status value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFOrmation message to be sent to control the specified feature indication. If the feature indicator is not associated with a call the specified B-channel should correspond to that associated with the data-link of the terminal. This will normally be the even B-channel. If the indicator is associated with a call, the B-channel of that call should be specified. A feature indication message is normally sent by a port configured as an NT (network termination).

Feature indicators can have a range of 0x0 to 0x3F. There is no universal standard as to the meaning of a particular feature indicator number, and different ISDN station sets will have varying numbers of indicators, some of which have fixed numbers and some of which can be programmed. The specific feature indicator number used will depend on the equipment and application.

Four status values are allowed which control the appearance of the indicator. The valid status characters and corresponding appearance are:

I	idle, off
A	active, solid illumination
P	pending, a slow blink
Q	prompt, a fast blink

The blinking rate will depend on the specific equipment used.

Example

xds_l3_feature_ind(1, 0x4, 0x3F, ‘P’);	this will send an INFOrmation message for B-channel 4 on board 1, for feature indicator 0x3F with a status of pending.

xds_l3_feature_key

xds_l3_feature_key(board_number, b_channel, feature);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int feature;	a value between 0x0 and 0x3F indicating the feature key

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message to indicate a feature key activation. Feature keys are used to notify the network for the purpose of controlling a feature.

Message Sent
"DFxxff" where **xx** is the B-channel number and **ff** is the feature key

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_FEATURE	invalid feature value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFOrmation message to be sent to control the specified feature. If a call is not active on the specified B-channel, the feature activation will be associated with the data-link that corresponds to that B-channel, otherwise, if a call is active, the feature key will be associated with the call. A feature key message is normally sent by a port configured as a TE (terminal equipment).

Feature keys can have a range of 0x0 to 0x3F. There is no universal standard as to the meaning of a particular feature key or feature key number. This will depend on the specifics of the switch and on any pre-subscribed features associated with the interface.

Example

xds_l3_feature_key(1, 0x4, 0x2);	this will send a INFOrmation message for B-channel 4 on board 1, for feature key 2.

xds_l3_hold

xds_l3_hold(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 HOLD message for the call currently associated with the specified B-Channel. A HOLD message is used to release the B-channel allocated to a call, but retain the call and its reference. A hold can be initiated by either the network or a terminal.

Message Sent
“DHxx” where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a HOLD message to be sent for the call associated with the specified B-channel. This can be done by either the network or a terminal. If the hold is accepted, a HOLD ACKnowledge message will be sent in response. This response message will include the call reference. It is the responsibility of the application to retain this call reference as it is needed to retrieve the call. A HOLD REJect message will be sent in response if the hold is not accepted. This will include the cause for rejection.

It should be noted that not all network switches support the HOLD feature. On others, it may be available by pre-subscription at the time the interface is ordered. Similarly, terminal equipment such as ISDN station sets may or may not support the feature.

Example

xds_l3_hold(1, 0x4);	this will send a HOLD message for B-channel 4 on board 1.

xds_l3_hold_ack

xds_l3_hold_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 HOLD_ACKnowledge message for the call currently associated with the specified B-Channel. A HOLD_ACK message is used to respond to a HOLD message and accept the release of the B-channel while retaining the call.

Message Sent
"DAxxA" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a HOLD ACKnowledge message to be sent for the call associated with the specified B-channel. This should be done in response to a HOLD message. The B-channel will be released for use with another call. The application should retain the call reference received in the hold message as it will be used when the call is retrieved.

Example

xds_l3_hold_ack(1, 0x4);	this will send a HOLD ACKnowledge message for B-channel 4 on board 1.

xds_l3_hold_rej

xds_l3_hold_rej(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the hold reject

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 HOLD REJect message for the call currently associated with the specified B-Channel. A HOLD REJect message is used to respond to a HOLD message and reject the release of the B-channel.

Message Sent
“DAxxRcc” where **xx** is the B-channel number and **cc** is the cause code

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a HOLD REJect message to be sent for the call associated with the specified B-channel. This should be done in response to a Hold message when the B-channel is not to be released. The cause value should indicate the reason for rejecting the hold. Typical reasons would be 0x45, “requested facility not implemented” if the application does not support hold, or 0x3, “requested facility not subscribed” when the hold feature is not enabled for a specific interface. For a list of cause codes and there meaning, see the XDS Layer 3 Protocol Software Manual, Q.850 or the relevant Bellcore documentation.

Example

xds_l3_hold_rej(1, 4, 0x45);	this will send a HOLD REJect message for B-channel 4 on board 1 with a cause of 0x45, “requested facility not implemented”.

xds_l3_info_request

xds_l3_info_request(board_number, b_channel, query, complete);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char query;	a character indicating the query type
char complete;	a 0x1 if the query is complete, otherwise 0

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message with an information request. This is done to query a terminal for information such as the SPID or address digits.

Message Sent
"DIxxQq" where **xx** is the B-channel number, and **q** is the information request type
"DIxxQqC" if the information is complete

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid query or complete value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This command is used to cause an INFOrmation message containing an information request. This should only be done on a port set to type NT if an S/T interface or to type LT if a U-Interface. Depending on the request type this may be for the terminal associated with the B- channel of for the call currently associated with the specified B-channel. The information will be returned in an INFOrmation message. It may be desirable to send a request complete message once the information and been received or upon a time out.
The valid information requests are:

A - Authorization Code
D - Address Digits
S - Terminal Identification (SPID)
U - Undefined

Example

xds_l3_info_request(1, 0x4, ‘S’, 0);	this will send a request for the SPID for B-channel 4 on board 1.

xds_l3_key_hold

xds_l3_key_hold(board_number, b_channel, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int reference;	a value between 0x00 and 0xFF that indicates the call reference of the call for which the notification is intended

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 KEY_HOLD message to inform an EKTS terminal that a call has been answered by another terminal.

Message Sent
"DExxHrr" where **xx** is the B-channel number and **rr** is the call reference.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	invalid call reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a KEY_HOLD message to be sent from an NT port to a Electronic Key Telephone Service (EKTS) terminal. The notification is used to inform the terminal that the call has been answered by another terminal. This is done so that the terminal may reflect the change of state by changing the state of an indicator (i.e. controlling whether the indicator blinks or not.)

Example

xds_l3_key_hold(1, 0x4, 0x1);	this will send a KEY_HOLD message for B-channel 4 on board 1 with a notification of Call Retrieved.

xds_l3_key_setup

xds_l3_key_setup(board_number, b_channel, bearer_cap, called_num, call_app);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char bearer_cap;	a character indicating the type of call, i.e. speech or data
char *called_num;	a pointer to an ASCII string with the called number
int call_app;	a value between 0x1 and 0xFE indicating the call appearance

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 KEY_SETUP message to an EKTS terminal to inform that terminal that another terminal is originating a call a call appearance or called number.

Message(s) Sent
For an EKTS port "DExxSb/#" where **xx** is the B-channel, **b** is the bearer capability and **/#** is the called party number.

For a CACH EKTS port "DExxSbA=ca" where **xx** is the B-channel, **b** is the bearer capability and **ca** is the call appearance number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_FEATURE	invalid feature value
ILL_ARG	an invalid bearer capability, call appearance value, or an invalid number format was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This command is used to cause a KEY_SETUP message to be sent to an Electronic Key Telephone Service (EKTS) terminal to indicate that another terminal in the EKTS group is originating a call on a Directory Number (DN) or call appearance. The DN format is used for Basic EKTS while the call appearance form is used for Call Appearance Call Handling (CACH) EKTS. If the CACH EKTS format is used, the called number may be set to a NULL string.

Example

xds_l3_key_setup(1, 0x4, ‘S’, ‘’, 0x3)	This will send a KEY_SETUP message for B-channel 4 on board 1 with a bearer capability of “speech” and a call appearance number of 3.

xds_l3_notify

xds_l3_notify(board_number, b_channel, notification, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char notification;	a character indicating the notification type
int reference;	a value between 0x00 and 0xFF that indicates the call reference of the call for which the notification is intended

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 NOTIFY message to inform an EKTS terminal of the change of state of a call.

Message Sent
"DExxNnrr" where **xx** is the B-channel number, **n** is the notification indicator, and **rr** is the call reference.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	invalid call reference value
ILL_ARG	an invalid notification value was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This command is used to cause a NOTIFY message to be sent from an NT port to a Electronic Key Telephone Service (EKTS) terminal. The notification is used to inform the terminal of a change of state of a call so that the terminal may reflect the change of state by changing the state of an indicator (i.e. controlling whether the indicator blinks or not.)
The valid notification indicators are:

B - Call Bridged
H - Call Held
M - Monitored User Idle
R - Call Retrieved

Example

xds_l3_notify(1, 0x4, ‘R’, 0x1);	this will send a NOTIFY message for B-channel 4 on board 1 with a notification of Call Retrieved.

xds_l3_proceed

xds_l3_proceed(board_number, b_channel, progress, signal);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the progress indicator
char signal;	a character indicating the signal if any

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 CALL PROCeeding message for the call currently associated with the specified B-Channel. A CALL PROCeeding message is used to notify the caller that all information necessary to process a call has been received and that call establishment has been initiated.

Message Sent
"DPxxps" where **xx** is the B-channel number, **p** is the progress indicator, and **s** is the signal

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid call progress indicator value
ILL_SIGNAL	an invalid signal value was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This command is used to cause a CALL PROCeeding message to be sent for the call associated with the specified B-channel. This should be done in response to a SETUP message for Enbloc sending, or when all necessary digits have been received for Overlap sending when the call can be initiated.

For ports that are configured as a TE (terminal equipment), the progress and signal values are not used. For an NT (network termination) port, the progress indicator may take values of:

I	Inband information or appropriate pattern now available
N	No progress indicator

The signal element is optional and may take the value ‘C’ for confirmation tone, or 0. If the signal element is indicates confirmation tone, the progress indicator should be set to ‘I’.

Example

xds_l3_proceed(1, 0x4, ‘N’, 0);	this will send a CALL PROCeeding message for B-channel 4 on board 1, with no progress indicator or signal.

xds_l3_progress

xds_l3_progress(board_number, b_channel, cause, progress, signal);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x0 and 0xff that indicates the reason for the PROGress message
char progress;	a character indicating the progress indicator
char signal;	a character indicating the signal if any

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 PROGress message for the call currently associated with the specified B-Channel. A PROGress message is used to notify the caller of interworking with non-ISDNs, a call is routed to an inband tone or announcement, or when a progress delay at the destination is reported.

Message Sent
"DPxxPccps" where **xx** is the B-channel number, **cc** is the cause value, **p** is the progress indicator, and s is the signal.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_PROGRESS	an invalid call progress value was used
ILL_SIGNAL	an invalid signal element was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a PROGress message to be sent for the call associated with the specified B-channel. This command is only valid for NT (network termination) ports.

A PROGress message is sent to report an interworking situation such as the destination is non ISDN, or a situation where the call, though initiated can not be completed, such as the called party is busy. In those cases, an inband signal such as busy tone is usually present and indicated by the progress indicator and signal elements in the message. An application would send a PROGress message when it realizes that a call can not be completed, or when interworking with the destination switch results in the receipt of a PROGress message from the destination.
The cause code is a value used to indicate the reason for the PROGress message. Examples would be a cause code of 0x11, “User busy”, or 0x12, “No user responding”. For a list of cause codes and there meaning, see the XDS Layer 3 Protocol Software Reference Manual, Q.850, or the relevant Bellcore documentation.
Progress values that are valid may be:

C	Call is not end to end ISDN, call progress information may be available inband
D	Destination address is non-ISDN
I	Inband information or appropriate pattern now available
W	Delay in response at destination interface

Valid Signal values are:

D	Dial tone
R	Audible ringback
N	Network Congestion/Reorder tone
B	Busy tone
C	Confirmation tone
F	Tones off

Example

xds_l3_progress(1, 0x4, 0x11, ‘I’, ‘B’);	this will send a PROGress message for B-channel 4 on board 1. The cause is 0x11, “User busy”, the progress indicator is inband information, and the signal is busy.

xds_l3_query_spid

xds_l3_query_spid(board_number, b_channel, spid_info);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
struct spid_dn *spid_info;	a structure to contain the SPID and DN information
{	
char spid[15];	an array to contain the SPID
char dn[8];	an array to contain the Directory Number
}	

Applicable boards
All XDS BRI boards

Purpose
This function is used to query the board to obtain the SPID and default Directory Number for a specific data-link.

Message Sent
"DQxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call
WRONG_QUERY	the wrong query response was received
WRONG_BOARD	a query response was received from the wrong board

Comments

This command is used to obtain the SPID (Service Profile Identifier) and default DN (directory number) associated with a specific data-link. Data-links are identified using numbers that take the same form as the B-channel numbers for a port. Each port may have one or two data-link and SPIDs associated with it. The data-link used for voice calls is associated with the even numbered B-channel and the data-link for data calls is associated with the odd numbered B-channel. If only one SPID and data-link is present, it will use the even number. Each B-channel has a default directory number that is used as the called party number unless otherwise specified in a SETUP message (see **xds_l3_setup).**

For ports configured as a TE (terminal equipment), the SPID and DN are programmed in by the application, and therefore, this command is only of use for diagnostic purposes, i.e. to see that the correct information has been programmed in. For NT (network termination) ports, the DN is programmed, but the SPID is received from the TEs connected to the port. This command may therefore be of use to determine which terminals are connected to which ports.
For the purposes of the XDS BRI Board, TE SPIDs are not used for call handling, but are used to associate TE equipment with data-links. The SPID programmed into the TE equipment must be at least 12 digits long. The 12th digit for the voice SPID must be a ‘1’, while the 12th digit for the data SPID must be a ‘2’. If two voice terminals are present, the second voice terminal must have a SPID where the 12th digit is ‘3’. If only one SPID is needed, it should have a 12th digit of ‘1’. The other digits are ignored by the board.

The default directory number, or DN, is a seven digit number in local ISDN format. It is used as the default called party number for calls placed to a TE by a port configured as an NT, unless otherwise specified. For a TE port, the DN is used as the calling number for calls placed using the specified B-channel.

Example

xds_l3_query_spid(0x1, 0x4, spid_buf)	This would request the SPID and DN for the B1 channel of the third port on board 1. The SPID and DN would be returned in the structure spid_buf.

xds_l3_query_status

xds_l3_query_status(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 STATus Enquiry message to the board. This can be done to prompt a terminal to respond with a STATUS message. This command is valid only for ports that are configured as an NT (network termination).

Message Sent
"DXxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to query an ISDN terminal about the call state of a call in a case where a received message indicates there may be a call processing discrepancy. The terminal, if it responds will send a STATUS message. As this response is dependent on the terminal and not on the board, this message will be returned on the receive message queue and will be retrieved with the **xds_msg_receive** function rather than the **xds_query_receive** function. This function is only valid for ports that are configured as an NT (network termination) port, and only for B-channels that have a call in process.

Example

xds_l3_query_status(0x1, 0x4)	This would send a STATus ENQuiry message for the B1 channel of the third port on board 1.

xds_l3_rel_com

xds_l3_rel_com(board_number, b_channel, cause, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the release
int reference;	a value between 0x00 and 0xFF that indicates the call reference for the call being released if it is does not have an B-channel allocated

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 REL_COMplete message for the call currently associated with the specified B-Channel or that is specified by the call reference. A REL_COMplete message is used to release a call reference and terminate a call in cases where a DISConnect message is inappropriate.

Message Sent
"DRxxccrr" where **xx** is the B-channel number **cc** is the cause value and **rr** is the optional call reference.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	an invalid call reference value was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This command is used to cause a REL_COMplete message to be sent for the call associated with the specified B-channel. This should be done to release a call and associated call reference under some circumstances. An example, is when a call has been placed on hold and is to be cleared, or when no TE equipment has responded to a SETUP message. If the call has been allocated a B-channel, a DISConnect should be sent to clear a call (see **xds_l3_disconnect**).
The cause code is a value used to indicate the reason for the REL_COMplete message. Examples would be a cause code of 0x10, “Normal clearing”, or 0x41, “Bearer capability not supported”. For a list of cause codes and there meaning, see the <u>XDS Layer 3 Protocol Software Manual</u>, Q.850, or the relevant Bellcore documentation.

In cases where the call is not currently allocated a B-channel, for example when on hold, it is necessary to use the call reference. This is a value in the range of 0x01 to 0x7F or 0x81 to 0xFF. This value should have been retained by the application from the received hold message. If the call reference is not implicitly specified, this value should be 0. The B channel in this case should correspond to the data-link of the call.

Example

xds_l3_rel_com(1, 0x4, 0x10, 0x02);	this will send a RELease COMplete message for B-channel 4 on board 1 with a cause value of 0x10, “Normal Clearing” and a call reference of 0x02.

xds_l3_retrieve

xds_l3_retrieve(board_number, b_channel, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int reference;	a value between 0x00 and 0xFF that indicates the call reference of the call being retrieved

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 RETrieve message for the call currently specified with the call reference. The call is to be allocated the specified B-channel, which must be available. A RETrieve message is used to reestablish a connection to a call placed on hold.

Message Sent
"DGxxrr" where **xx** is the B-channel number and **rr** is the call reference

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	an invalid call reference value was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This command is used to cause a RETrieve message to be sent for the call with the specified reference. The call will be allocated the specified B-channel. This is done when retrieving a call previously placed on hold using a HOLD message. Note that while both the NT (network termination) and TE (terminal equipment) side of a connection can place a call on hold, only the TE user can retrieve a call. Also, the selected B-channel must be available.

The NT side should respond with either a RETrieve ACKnowledge message, in which case, the connection is restored, or a RETrieve REJect message, in which case, the call remains in the held state.

The call reference used in this message is the one given in the original HOLD or HOLD ACKnowledge message. It is the responsibility of the application to retain this reference as long as the call is in the hold state. Failure to do so may result in irretrievable or un-releasable calls.

Example

xds_l3_retrieve(1, 0x4, 0x83);	this will send a RETreive message for B-channel 4 on board 1 with a call reference of 0x83.

xds_l3_retrieve_ack

xds_l3_retrieve_ack(board_number, b_channel, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int b_channel;	the number of the B-channel that indicates the BRI interface
int reference;	a value between 0x00 and 0xFF indicating the call being retrieved

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 RETrieve ACKnowledge message for the call specified by the call reference. The call will be allocated the specified B-channel. A RETrieve ACKnowledge message is used to grant a RETrieve request.

Message Sent
"DGxxArr" where **xx** is the B-channel number and **rr** is the call reference number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	an invalid call reference value was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a RETrieve ACKnowledge message to be sent for the specified call reference. The call will be allocated the specified B-channel. Note, that this channel must match that received in the RETrieve message. If a RETrieve ACKnowledge message is sent, the connection to the call is reestablished on the specified B channel. Note, that as only a TE (terminal equipment) port can send a retrieve message, only an NT (network termination) port can grant the retrieval by sending a RETrieve ACKnowledge message.

Example

xds_l3_retrieve_ack(1, 0x4, 0x3);	this will send a RETrieve ACKnowledge message for B-channel 4 on board 1, with call reference of 0x3.

xds_l3_retrieve_rej

xds_l3_retrieve_rej(board_number, b_channel, cause, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for rejecting the retrieval
int reference;	a value between 0x00 and 0xFF that indicates the call for which retrieval is being rejected

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 RETrieve REJect message for the call specified by the call reference. A RETrieve REJect message is used to deny a RETrieve request message.

Message Sent
"DGxxRccrr" where **xx** is the B-channel number, **cc** is the cause, and **rr** is the call reference.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	an invalid call reference value was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a RETrieve REJect message to be sent for the call specified by the call reference value. The B-channel should match that in the received RETrieve message. Note, that as only a TE (terminal equipment) port can send a retrieve message, only an NT (network termination) port can grant the retrieval by sending a RETrieve REJect message.
The cause code is a value used to indicate the reason for the PROGress message. Examples would be a cause code of 0x06, “Channel unacceptable”, or 0x2A, “Switching equipment congestion”. For a list of cause codes and there meaning, see the XDS Layer 3 Protocol Software Reference Manual, Q.850, or the relevant Bellcore documentation.

Example

xds_l3_retrieve_rej(1,0x4,2A,0x03);	this will send a RETrieve REJect message for B-channel 4 on board 1, with a cause of 0x2A, “Switching equipment congestion”, and a call reference of 0x3.

xds_l3_setup

xds_l3_setup(board_number, b_channel, port_typ, bearer_cap, progress, signal, called_num, calling_num, call_app);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating whether the port is of type NT or TE
char bearer_cap;	a character indicating the type of call, i.e. speech or data
char progress;	a character indicating the progress indicator
char signal;	a character indicating the alerting signal if any
char *called_num;	a pointer to an ASCII string with the called number
char *calling_num;	a pointer to an ASCII string with the calling number
int call_app;	a value between 0x1 and 0xFE indicating the call appearance

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 SETUP message. A SETUP message is used to initiate a call. As the SETUP message takes a different form for NT and TE interface, the presence of some arguments will depend on the port type.

Message Sent
For an NT interface: "DSxxbps(calling#/called#)", where **xx** is the B-channel number, **b** is the bearer capability, **p** is the progress indicator, **s** is the signal, **calling#** is the directory number of the calling party and optionally **called#** is the number being called.

For an NT interface to CACH EKTS terminals: "DSxxbps(calling#)A=ca" where **ca** is the two digit call appearance number in hex.

For a TE interface: "DSxxbcalled#", where **xx** is the B-channel number, **b** is the bearer capability, **called #** is the number being called.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_TYPE	invalid port type
ILL_ARG	an invalid bearer capability, call appearance number, or an invalid calling or called number format was used
ILL_SIGNAL	invalid signal value
ILL_PROGRESS	an invalid call progress indicator value was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This command is used to cause a SETUP message to be sent to initiate a call. The board level command takes a different form depending on whether the port is configured as an NT (network termination) or a TE (terminal equipment). The progress and signal elements are not present in the SETUP message for a TE, and the treatment of the directory numbers is different. The port type must be specified by the **port_type** argument. Valid values are 'N' for an NT interface and 'T' for a TE interface. Note, that for a U-Interface board, the 'N' will refer to a Line Termination or LT port type and the 'T' will refer to a Network Termination or NT port type.

The XDS and Infinity BRI Boards uses exclusive B channel allocation only. If the B-channel specified for the SETUP message is in use at the time the message is sent, the channel ID will be for no B-channel. This should only be used for ports of type 'N'.

The bearer capability indicates the type of call being made, i.e. voice or data. There are four valid choices:

A	3.1 kHz. audio (used for voice and modems)
D	unrestricted digital information, 64 kbps
R	rate adapted data, rate adapted for 56 kbps
S	speech (suitable for voice, but not modems)

For NT ports, a progress indicator must be specified. This argument is used in interworking situations where the call is not completely ISDN. The valid values are:

C	Call is not end-to-end ISDN
O	Originating address is non-ISDN
N	No progress indicator

Note that if there are no interworking problems, the progress indicator must be set to 'N'.
The signal argument is used to indicate which type of alerting is to be used by the terminal. This argument is used only for NT ports. Valid values are:

N	Normal alerting
D	Distinctive alerting
S	Special alerting
I	Intercom alerting
R	Reminder ring
F	Alerting off

It should be noted that there is no uniform format for the various forms of alerting. Normal ringing typically sounds like the standard 2 on 4 off ring of an analog phone, but the other alerting codes will produce different patterns of tones or beeps depending on the specific piece of terminal equipment.
For TE ports, the called number is the directory number of the destination. The calling number

element will use the default DN of the B-channel on which the call is being placed. The called number can be up to 15 digits long.

For NT ports, only the calling number is required. If the called number is a NULL string, the default DN of the B-channel will be used as the called number element, otherwise, the called number specified will be used. The calling number can be up to 15 digits long. If a called number is specified, the total number of digits in the called and calling number is restricted to 24. This should not be a problem as a 7 digit called number is normally sufficient.
The call_app value is used for CACH EKTS calls only for an NT port. For non-CACH EKTS calls, this value should be 0.

Example

For an NT port

xds_l3_setup(1, 0x4, ‘N’, ‘S’, ‘N’, ‘N’, ‘’, “5551212”, 0);	this will send a SETUP message for B-channel 4 on board 1 with a bearer capability of speech, no progress indicator, normal alerting, the default called number and calling number of 555-1212.

For an NT CACH EKTS port

xds_l3_setup(1, 0x4, ‘N’, ‘S’, ‘N’, ‘N’, ‘’, “5551212”, 3);	this will send a SETUP message for B-channel 4 on board 1 with a bearer capability of speech, no progress indicator, normal alerting, a calling number of 555-1212, and call appearance of 3.

For a TE port

xds_l3_setup(1, 0x4, ‘T’, ‘D’, “18005551212”, ‘’, 0);	this will send a SETUP message for B-channel 4 on board 1 with a bearer capability of data and a called number of 1-800-555-1212.

xds_l3_spid

xds_l3_spid(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message containing the SPID for the data-link associated with the specified B-Channel. The SPID (Service Profile Identifier) is used by the network switch to identify a particular piece of terminal equipment.

Message Sent
"DIxxS" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFOrmation message to be sent containing the SPID for the data-link associated with the specified B-channel. The SPID must first be programmed into the board using the function **xds_set_bri_dn.** This information can be stored in the EEAROM on the board using the **xds_set_config** function so that it does not have to be reloaded every time the computer is rebooted.

This command is valid only for ports configured as TE (terminal equipment). If only one SPID is required on the port, it should be sent for the even B-channel. If two SPIDs are required, it should be sent for both B-channels.

A SPID must be sent to the network before calls can be placed or received. The data-link must first be established and a TEI assigned. This normally takes place automatically within a few seconds of power up. A TEI assigned message will be received for each data-link on the port.

Example

xds_l3_spid(1, 0x4);	this will send an INFOrmation message for B-channel 4 on board 1 containing the SPID.

xds_l3_text

xds_l3_text(board_number, b_channel, mode, l3_text);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char mode;	a value between 0x0 and 0x4 indicating the text operation
char *l3_text	a pointer to a NULL terminated string of up to 20 ASCII characters

Applicable boards
All XDS BRI boards

Purpose
This function is used to prepare text to be included in a Layer 3 message. Several modes are available that allow inclusion of text into a SETUP message, or provide for sending an INFOrmation message with text. Only NT (network termination) ports can send text.

Message Sent
"DTxxC0" where **xx** is the B-channel, clears the buffer
"DIxxTtext" where **xx** is the B-channel number to send a single line of **text**
"DTxxC1" where **xx** is the B-channel and line 1 is the line to clear
"DTxxC2" where **xx** is the B-channel and line 2 is the line to clear
"DIxxL1text" where **xx** is the B-channel and line 1 is the line for the **text** to be sent
"DIxxL2text" where **xx** is the B-channel and line 2 is the line for the **text** to be sent
"DIxxB" where **xx** is the B-channel to send the buffer

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	text message is too long
ILL_MODE	invalid call forwarding mode
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to prepare text to send in a Layer 3 message. Display text is arranged in lines of up to 20 ASCII characters. Text can be sent immediately, or it can be put in a buffer with 2 lines. Text can either be sent as an INFOrmation message, or as part of a SETUP message. The "mode" controls which text operation is carried out. The valid mode values are:

0x0	clear the buffer
0x1	send the text in the buffer as an INFOrmation message
0x2	send a line of text immediately as an INFOrmation message
0x3	set line 1 of the buffer
0x4	set line 2 of the buffer
0x5	clear line 1 of the buffer
0x6	clear line 2 of the buffer

Each B-channel has its own two-line buffer. If a SETUP command is sent, it will include any text in the buffer as an information element. The text buffer is cleared whenever a RELease COMplete message is sent or received by the board.

If a call is allocated a B-channel, then any INFOrmation message containing text will be sent with the call reference for that call, and the terminal will associate the text with the call. If the B-channel is not allocated to a call, the INFOrmation message will not have a call reference. The even B-channel will send text using the data-link associated with the 1st SPID, and the odd B-channel will use the data-link associated with the 2nd SPID. Where a terminal has two SPIDs, one for voice and one for data, the first SPID will normally be used. Most terminals will display the text. However, at least some terminals will clear the display after a few seconds.

Example

xds_l3_text(1, 0x4, 0, ‘’);	this will clear the text buffer for B-channel 4 on board 1.
xds_l3_text(1, 0x4, 1, "This is text");	this will send the text "This is text" for B channel 4 on board 1.
xds_l3_text(1, 0x4, 3, "This is line 1");	will set line 1 of the buffer for B channel 4 on board 1 to "This is line 1"
xds_l3_text(1, 0x4, 2, ‘’);	this will send the contents of the buffer as an INFOrmation message for B-channel 4 on board 1.

xds_l3_message

int xds_l3_message(board_number, port, sapi, tei, type, l3MsgLen, l3message);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating the port to control
int sapi;	a value between 0x0 and 03F indicating the SAPI to be used
int tei;	a value between 0x0 and 0x7F indicating the TEI to be used
char type;	either 'C' for command or 'R' for response
unsigned short l3MsgLen;	the number of octets in the Layer 3 message
unsigned char *l3message;	a pointer to an array containing the Layer 3 message

Applicable boards
All XDS BRI boards

Purpose
This function is used to send a Layer 3 message using the DLCI specified by the SAPI and TEI specified on the BRI interface specified by the port argument. The message is contained in the array pointed to by *message and has a length in octets of l3MsgLen.

Message Sent
"LCxxyyzz" where **C** is the message type, **xx** is the BRI interface number**,** **yy** is the SAPI and **zz** is the TEI number. The Layer 3 message and length is placed in the auxiliary mailbox.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_TYPE	invalid message type
ILL_ARG	the TEI specified is outside the range 0x0 to 0x7E, or the SAPI is outside the range 0x0 to 0x3F.
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a Layer 3 message using the data link specified by the DLCI (Data Link Connection Identifier) composed of the SAPI (Service Access Point Identifier) and TEI (Terminal Endpoint Identifier). The SAPI has a range of 0-63 while the TEI has a range of 0-127. The normal rules for allowed DLCI addresses apply, and the board currently only supports DLCIs of [63, 127] and in the range [0, 0-127] and [16, 0-126]. The Layer 3 message itself consists of 1 to 260 octets. The contents of the message are not restricted to a Q.931 format. The TEI must already be assigned to the interface if it is in the range 0-126. The broadcast TEI of 127 is always available, and will result in an unnumbered information frame being sent. Two types of messages are allowed, commands and responses, specified by a type of ‘C’ or a ‘R’. The normal type should be ‘C’ or command, and Q.931 specifies only commands.

Example

xds_l3_message(0, 2, 0, 0x40, ‘C’, 4, cnctmsg);	this will send a Layer 3 message 4 octets in length with a SAPI of 0 and a TEI of 64 on port 2 of board 0. In this case, cnctmsg is an array of four octets that is a Q.931 connect message.
	cnctmsg[4] = {0x8, 0x1, 0x81, 0x7};

Basic Rate EuroISDN Layer 3 ‘D’ Functions

This page was intentionally left blank.

xds_euro_l3_alert

xds_euro_l3_alert(board_number, b_channel, progress);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the progress indicator if any

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 ALERTing message for the call currently associated with the specified B-Channel. An ALERTing message is used to notify the caller that the called party is being informed of an incoming call.

Message Sent
"DAxxp" where **xx** is the B-channel number, and **p** is the progress indicator.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an ALERTing message to be sent for the call associated with the specified B-channel. If the port is configured as an NT (network termination), progress indication and signal information elements can be sent as part of the message.

Valid values for the progress indicator are:

C	Call not end-to-end ISDN
D	Destination not ISDN
I	Inband information or appropriate pattern now available
N	No progress indicator

Example

xds_euro_l3_alert(1, 0x4, 'I');	this will send an ALERTing message for B-channel 4 on board 1. The progress indicator is set for inband signal.

xds_euro_l3_call_forwarding

xds_euro_l3_call_forwarding(board_number, b_channel, mode, id, operation, procedure, service, forwarding_num, subscriber_num, error_code)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char mode;	a value indicating the command mode to be used
int id;	the invoke id value
char operation;	a value indicating which operation to be used
char procedure;	a value indicating which procedure to be used
char service;	a value indicating which basic service to be used
char *forwarding_num;	a pointer to an ASCII string with the number to be forwarded
char *subscriber_num;	a pointer to an ASCII string with the subscriber number
int error;	the value of an error code

Applicable boards
All EuroISDN BRI Boards

Purpose
This function is used to send a Layer 3 FACILITY message to control the call forwarding. It takes on one of three different forms, described below.

Message Sent

"DFxxFIidopb#/#"	Invoke mode, where **xx** is the b-channel, **id** is the invoke id value, **o** is the operation mode, **p** is the procedure mode, **b** is the basic service mode, the **#** is the number forwarded to, and **/#** is the subscriber number
"DFxxFRid"	Return Result mode, where **xx** is the b-channel, **id** is the invoke id value
"DFxxFEidee"	Return Error mode, where **xx** is the b-channel, **id** is the invoke id value, **ee** is the error value

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid ID tag code, operation value, procedure value, called number, or calling number
ILL_MODE	invalid call forwarding mode
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause various messages to be sent according to how the function is called.

The legal values for **mode** are ‘I’ = invoke, ‘R’ = return results, and ‘E’ = return error. The legal values for id are from 0 to 0xFF. The legal values for **operation** can be ‘F’ = call forwarding off, ‘N’ = call forwarding on, and ‘Q’ = call forwarding interrogate. The legal **procedure** values are ‘A’ = call forward always CFU, ‘B’ = call forward busy CFB, and ‘N’ = call forward on answer CFNR. The **basic service** values are as follows: ‘A’ = 1, ‘B’ = 3, ‘C’ = 20h, and ‘D’ = others, 0.

Examples
xds_euro_l3_call_forwarding(1, 0x4, ‘I’, 0x3F, ‘F’, ‘B’, ‘A’, ‘9’, ‘3’);

this will send an invoke message for B-channel 4, on board 1, an invoke id of 3F, call forwarding off ‘F’, call forward busy CFB ‘B’, basic service value A ‘A’, ‘9’ is the forwarded number, and ‘3’ is the subscriber number.

xds_euro_l3_call_forwarding(1, 0x4, ‘R’, 0x3F);

this will send a return result message for B-channel 4, on board 1, and an invoke id of 3F

xds_euro_l3_call_forwarding(1, 0x4, ‘E’, 0x3F, 0x93);

this will send a return error message for B-channel 4, on board 1, an invoke id of 3F, and a value of 93 for the error

xds_euro_l3_called_num

xds_euro_l3_called_num(board_number, b_channel, complete, called_num);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char complete;	a value indicating whether sending complete element should be included
char *called_num;	a pointer to an ASCII string with the called number

Applicable boards
All EuroISDN BRI Boards

Purpose
This function is used to send a Layer 3 INFORMATION message containing called party number information for the call currently associated with the specified B-Channel. This is done as part of the call setup procedure when using overlap sending.

Message Sent
"DKxxk" where **xx** is the B-channel number and **k** is the called party number digit
"DKxxC" where the sending complete element is to be sent (complete = 0x1)

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This command is used to cause an INFORMATION message containing a called party number element to be sent for the call associated with the specified B-channel. This should be done as part of the overlap sending procedure. The function **xds_euro_l3_setup** should first be called with a NULL called number string. Typically, only the user or terminal side of a BRI interface will use overlap sending. One or more digits may be sent in each message. If the value of complete is 0x1, an INFORMATION message with the Sending Complete element will be sent. This may be done after the last digit of the called party number has been sent, however, it is not mandatory.

Example

xds_euro_l3_called_num(1, 0x4, 0, ‘3’);	this will send an INFORMATION message for B-channel 4 on board 1 with a called party number digit of ‘3’.

xds_euro_l3_connect

xds_euro_l3_connect(board_number, b_channel, progress);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the progress indicator if any

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 CONNect message for the call currently associated with the specified B-Channel. A CONNect message is used to notify the caller that the called party has answered an incoming call.

Message Sent
“DCxx” where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a CONNect message to be sent for the call associated with the specified B-channel. This should be done when the called party answers.

Valid values for the progress indicator are:

C	Call not end to end ISDN
D	Destination not ISDN
I	Inband information or appropriate pattern now available
O	Origination not ISDN
R	Return to ISDN
N	No progress indicator

Example

xds_euro_l3_connect(1, 0x4, ‘N’);	this will send a CONNect message for B-channel 4 on board 1 with no progress indicator.

xds_euro_l3_disconnect

xds_euro_l3_disconnect(board_number,b_channel,cause,progress,reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the disconnect
char progress;	a character indicating the progress indicator if any
int reference;	a value between 0x00 and 0xFF that is the call reference number

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 DISConnect message for the call currently associated with the specified B-Channel. A DISConnect message is used to notify either the network or the user that a disconnect sequence has been initiated.

Message Sent
“DDxxccprr” where **xx** is the B-channel number and **cc** is the cause code, **p** is the progress indictor and **rr** is the optional call reference

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	the cause value is invalid
ILL_PROGRESS	invalid progress indicator value
ILL_FEATURE	invalid feature value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a DISConnect message to be sent for the call associated with the specified B-channel. This should be done when initiating a disconnect sequence.

The cause code is a value used to indicate the reason for a particular message, in this case, why a disconnect is occurring. A cause value of 0x10 indicates normal clearing. Other values may indicate error conditions. For a list of cause codes and their meanings, see the appropriate Basic Rate ISDN Board Technical Manual (EuroISDN Version), Q.850 or ETS 300 102.

If a tone or announcement is to be played, the progress should be ‘I’, otherwise, this should be ‘N’. If the call to be released is associated with a B-channel, the call reference specified should be 0x00.

Example

xds_euro_l3_disconnect(1, 0x4, 0x10, ‘N’, 0);	this will send a DISConnect message for B-channel 4 on board 1, with a cause of 0x10 (normal clearing).

xds_euro_l3_facility

xds_euro_l3_facility(board_number, b_channel, component, reference, id_tag, tag_value)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char component;	'I', 'E', 'R', or 'J'
int reference;	a value between 0x00 and 0xFF that is the call reference number
int id_tag	a value between 0x00 and 0x7F that is the id tag number
int tag_value	a value between 0x00 and 0xFF that is the operation or error tag value

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 EURO ISDN Facility message.

Message Sent
"DFxxcrrid(tt)" where **xx** is the B-channel number, **c** is the component selected, **rr** is the call reference number, **id** is the id tag number, and **tt** is the operation or error tag number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid component or ID tag
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a Layer 3 FACILITY message to the board. The legal values for component are: ‘I’ = invoke, ‘E’ = return error, ‘R’ = return results, and ‘J’ = reject.

Example

xds_euro_l3_facility(1, 0x4, ‘I’, 0x7F, 0x19, 0x11);	this will send a FACILITY message for B-channel 4, on board 1, using the invoke ‘I’ component, a call reference of 0x7F, an id tag value of 0x19, and operation/error tag value of 0x11.

xds_euro_l3_hold

xds_euro_l3_hold(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 HOLD message to a B-channel.

Message Sent
"DHxx" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD message to a selected b-channel on a selected board.

Example

xds_euro_l3_hold(1,0x4);	this sends a message (B-channel 4 to board 1)

xds_euro_l3_hold_ack

xds_euro_l3_hold_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 HOLD ACKnowledge message to a B-channel.

Message Sent
"DHxxA" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD ACKnowledge message to a selected b-channel on a selected board.

Example

xds_euro_l3_hold_ack(1, 0x4);	this sends a message (B-channel 4 to board 1)

xds_euro_l3_hold_rej

xds_euro_l3_hold_rej(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the hold rejection

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 HOLD REJect message to a B-channel.

Message Sent
“DHxxRcc” where **xx** is the B-channel number, and cc is the cause for the rejection

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD REJect message to a selected b-channel on a selected board.

Example

xds_euro_l3_hold_rej(1, 0x4, 0x7F);	this sends a message (B-channel 4 to board 1, cause 0x7FF)

xds_euro_l3_info

xds_euro_l3_info(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value for the cause element

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message for the call currently associated with the specified B-Channel containing a cause element for the call.

Message Sent
"DIxxcc" where **xx** is the B-channel number and **cc** is the cause value

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause element value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send an INFOrmation message to be sent for the call associated with the specified B-channel.

Example

xds_euro_l3_info(1, 0x4, 0x1F);	this will send an INFOrmation message for B-channel 4 on board 1 with a cause element value of 0x1F.

xds_euro_l3_key_fac

xds_euro_l3_key_fac(board_number, b_channel, cause, key_fac);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the disconnect
char *key_fac	a pointer to an ASCII string with any keypad facility digits

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message with a Keypad Facility element to control supplemental services.

Message Sent
"DIxxKkk" where **xx** is the B-channel number and **kk** are the Keypad digits, or
"DIxxccKkk" where **cc** is a optional cause value.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a INFORMATION message with a Keypad Facilities element to be sent. An optional cause element may be included if the cause value is non-zero. The Keypad Facilities element is used to control supplemental services which are specific to the particular switch and network. The Keypad Facilities element consists of one or more keypad digits as found on a telephone set keypad, typically in the range 0-9, *, and #.

Example

xds_euro_l3_key_fac(1, 0x4, 0, "2*");	this sends a message (B-channel 4 to board 1with Keypad Facility string of "2*", no cause element)

xds_euro_l3_notify

xds_euro_l3_notify(board_number, b_channel, notification);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char notification;	a character indicating the notification type

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 NOTIFY message to inform a user or terminal of the change of state of a call due to call rearrangement.

Message Sent
"DNxxn" where **xx** is the B-channel number and n is the notification indicator.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid notification value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a NOTIFY message to be sent from an NT port to a user or terminal. The notification is used to inform the terminal of a change of state of a call due to call rearrangement so that the terminal may reflect the change of state by changing the state of an indicator (i.e. controlling whether an indicator blinks or not) or other means of informing the user.
The valid notification indicators are:
R - Call Resumed
S - Call Suspended

Example

xds_euro_l3_notify(1, 0x4, 'R');	this will send a NOTIFY message for B-channel 4 on board 1 with a notification of Call Resumed.

xds_euro_l3_proceed

xds_euro_l3_proceed(board_number, b_channel, progress);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the progress indicator

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 CALL PROCeeding message for the call currently associated with the specified B-Channel. A CALL PROCeeding message is used to notify the caller that all information necessary to process a call has been received and that call establishment has been initiated.

Message Sent
"DPxxp" where **xx** is the B-channel number and p is the progress indicator.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a CALL PROCeeding message to be sent for the call associated with the specified B-channel. This should be done in response to a SETUP message for Enbloc sending, or when all necessary digits have been received for Overlap sending when the call can be initiated.

The progress indicator may take values of:

C	Call not end to end ISDN
D	Destination not ISDN
I	Inband information or appropriate pattern now available
O	Origination not ISDN
R	Return to ISDN
N	No progress indicator

Example

xds_euro_l3_proceed(1, 0x4, ‘C’);	this will send a CALL PROCeeding message for B-channel 4 on board 1, with a progress indicator of call not end to end ISDN.

xds_euro_l3_progress

xds_euro_l3_progress(board_number, b_channel, cause, progress);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x0 and 0xff that indicates the reason for the PROGress message
char progress;	a character indicating the progress indicator

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 PROGress message for the call currently associated with the specified B-Channel. A PROGress message is used to notify the caller of interworking with non-ISDNs, a call is routed to an inband tone or announcement, or when a progress delay at the destination is reported.

Message Sent
"DPxxPccp" where **xx** is the B-channel number, **cc** is the cause value, and **p** is the progress indicator.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_PROGRESS	invalid progress indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a PROGress message to be sent for the call associated with the specified B-channel. A PROGress message is sent to report an interworking situation such as the destination is non ISDN, or a situation where the call, though initiated can not be completed, such as the called party is busy. In those cases, an inband signal such as busy tone is usually present and indicated by the progress indicator and signal elements in the message. An application would send a PROGress message when it realizes that a call can not be completed, or when interworking with the destination switch results in the receipt of a PROGress message from the destination.

The cause code is a value used to indicate the reason for the PROGress message. Examples would be a cause code of 0x11, ‘User busy’, or 0x12, ‘No user responding’. For a list of cause codes and their meaning, see the appropriate XDS Basic Rate ISDN Board Technical Manual (EuroISDN Version), Q.850, or ETS 300 102.
Progress values that are valid may be:

C	Call is not end-to-end ISDN
D	Destination is non-ISDN
I	Inband information or appropriate pattern now available
O	Origination is non-ISDN
R	Return to ISDN
N	no progress indicator

Example

xds_euro_l3_progress(1, 0x4, 0x11, ‘I’);	this will send a PROGress message for B-channel 4 on board 1. The cause is 0x11, ‘User busy’, the progress indicator is inband information.

xds_euro_l3_query_dn

xds_euro_l3_query_dn(board_number, b_channel, dn_info);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
struct dn *dn_info;	a structure to contain the DN information
{	
char dn[16];	an array to contain the Directory Number
}	

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to query the board to obtain the default Directory Number for a specific B-channel.

Message Sent
"DQxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
WRONG_BOARD	a query response was received from the wrong board
WRONG_QUERY	the wrong query response was received
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to obtain the default DN (directory number) or subscriber number associated with a specific B-channel. Each B-channel has a default directory or subscriber number that is used as the calling party number for TE ports and the called party number for NT ports in SETUP messages (see **xds_euro_l3_setup)** unless otherwise specified**.** This number may contain up to 15 digits. The DN is programmed in by the application, and therefore, this command is only of use for diagnostic purposes, i.e. to see that the correct information has been programmed in.

Example

xds_euro_l3_query_dn(0x1, 0x4, dn_buf)	This would request the DN for the B1 channel of the third port on board 1. The DN would be returned in the structure dn_buf.

xds_euro_l3_query_status

xds_euro_l3_query_status(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 STATus Enquiry message to the board. This can be done to prompt a terminal to respond with a STATUS message.

Message Sent
"DXxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to query an ISDN terminal about the call state of a call in a case where a received message indicates there may be a call processing discrepancy. The terminal, if it responds will send a STATUS message. As this response is dependent on the terminal and not on the board, this message will be returned on the receive message queue and will be retrieved with the **xds_msg_receive** function rather than the **xds_query_receive** function. This function is only valid for B-channels that have a call in process.

Example

xds_euro_l3_query_status(0x1, 0x4)	This would send a STATus ENQuiry message for the B1 channel of the third port on board 1.

xds_euro_l3_rel_com

xds_euro_l3_rel_com(board_number, b_channel, cause, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the release
int reference;	a value between 0x00 and 0xFF that indicates the call reference for the call being released if it is not allocated a B-channel

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 REL_COMplete message for the call currently associated with the specified B-Channel or that is specified by the call reference. A REL_COMplete message is used to release a call reference and terminate a call in cases where a DISConnect message is inappropriate.

Message Sent
"DRxxccrr" where **xx** is the B-channel number **cc** is the cause value and **rr** is the optional call reference.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	invalid reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a REL_COMplete message to be sent for the call associated with the specified B-channel. This should be done to release a call and associated call reference under some circumstances. An example is when no TE equipment has responded to a SETUP message. If the call has been allocated a B-channel, a DISConnect should be sent to clear a call (see **xds_l3_disconnect**).

The cause code is a value used to indicate the reason for the REL_COMplete message. Examples would be a cause code of 0x10, ‘Normal clearing’, or 0x41, ‘Bearer capability not supported’. For a list of cause codes and their meaning, see the appropriate XDS Basic Rate ISDN Board Technical Manual (EuroISDN Version), Q.850, or ETS 300 102.
If the call to be released is associated with a B-channel, the call reference specified should be 0x00.

Example

xds_euro_l3_rel_com(1, 0x4, 0x10, 0x00);	this will send a RELease COMplete message for B-channel 4 on board 1 with a cause value of 0x10, “Normal Clearing” and no call reference.

xds_euro_l3_resume

xds_euro_l3_resume(board_number, b_channel, call_id);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *call_id;	a pointer to an ASCII string used to identify the suspended call

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 RESUME message for the call specified by the call identity. A RESUME message is used to reestablish a connection to a suspended call.

Message Sent
"DGxxS(id)" where **xx** is the B-channel number and id is the call identity string.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a RESUME message to be sent for the call with the specified call identity. The call will be allocated the specified B-channel. This is done when resuming a call previously suspended using a SUSPEND message. The selected B-channel must be available. The NT side should respond with either a RESUME ACKnowledge message, in which case, the connection is restored, or a RESUME REJect message, in which case, the call remains in the suspended state.

The call identity used in this message is the one used in the original SUSPEND message. It is the responsibility of the application to retain this call identity as long as the call is in the suspended state. Failure to do so may result in irretrievable or unreleasable calls.

Example

xds_euro_l3_resume(1, 0x4, 'Call #1');	this will send a RESUME message for B-channel 4 on board 1 with a call identity of 'Call #1'

xds_euro_l3_resume_ack

xds_euro_l3_resume_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 RESUME ACKnowledge message in response to a RESUME message. The call will be allocated the specified B-channel.

Message Sent
“DGxxA” where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a RESUME ACKnowledge message to be sent in response to a RESUME message. The call will be allocated the specified B-channel. Note, that this channel must match that received in the RESUME message. If a RESUME ACKnowledge message is sent, the connection to the call is reestablished on the specified B channel. The Call Identity of the resumed call is then free. Note, that as only a TE (terminal equipment) port can send a RESUME message, only an NT (network termination) port can grant the resumption by sending a RESUME ACKnowledge message.

Example

xds_euro_l3_resume_ack(1, 0x4);	this will send a RESUME ACKnowledge message for B-channel 4 on board 1.

xds_euro_l3_resume_rej

xds_euro_l3_resume_rej(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for rejecting the resumptions

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 RESUME REJect message for the call on the specified B-channel. A RESUME REJect message is used to deny a RESUME request message.

Message Sent
"DGxxRcc" where **xx** is the B-channel number and **cc** is the cause.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a RESUME REJect message to be sent for the call on the specified B-channel. The B-channel should match that in the received RESUME message. Note, that as only a TE (terminal equipment) port can send a RESUME message, and only an NT (network termination) port can grant the retrieval by sending a RESUME REJect message.
The cause code is a value used to indicate the reason for the PROGress message. Examples would be a cause code of 0x06, 'Channel unacceptable', or 0x2A, 'Switching equipment congestion'. For a list of cause codes and their meaning, see the appropriate XDS Basic Rate ISDN Board Technical Manual (EuroISDN Version), Q.850, or ETS 300 102.

Example

xds_euro_l3_resume_rej(1, 0x4, 0x2A);	this will send a RESUME REJect message for B-channel 4 on board 1, with a cause of 0x2A, 'Switching equipment congestion'.

xds_euro_l3_retrieve

xds_euro_l3_retrieve(board_number, b_channel, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int reference;	a value between 0x0 and 0xFF that indicates the reference number of the call

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 RETrieve message to a B-channel.

Message Sent
"DGxxrr" where **xx** is the B-channel number, and **rr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	invalid reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a RETrieve message to a selected b-channel on a selected board.

Example

xds_euro_l3_retrieve(1, 0x4, 0x93);	this sends a message (B-channel 4 to board 1 with a call reference # of 0x93)

xds_euro_l3_retrieve_ack

xds_euro_l3_retrieve_ack(board_number, b_channel, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int reference;	a value between 0x0 and 0xFF that indicates the reference number of the call

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 RETrieve ACKnowledge message to a B-channel.

Message Sent
“DGxxArr” where **xx** is the B-channel number, and **rr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	invalid reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a RETrieve ACKnowledge message to a selected b-channel on a selected board.

Example

xds_euro_l3_retrieve_ack(1, 0x4, 0x93);	this sends a message (B-channel 4 to board 1 with a call reference # of 0x93)

xds_euro_l3_retrieve_rej

xds_euro_l3_retrieve_rej(board_number, b_channel, cause, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x0 and 0xFF that indicates the cause value of the rejection
int reference;	a value between 0x0 and 0xFF that indicates the reference number of the call

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 RETrieve REJect message to a B-channel.

Message Sent
“DGxxArr” where **xx** is the B-channel number, and **rr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	invalid reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a RETrieve REJect message to a selected b-channel on a selected board.

Example

xds_euro_l3_retrieve_ack(1, 0x4, 0x19, 0x93);	this sends a message (B-channel 4 to board 1 with a cause value of 0x19 and a call reference # of 0x93)

xds_euro_l3_setup

xds_euro_l3_setup(board_number, b_channel, port_type, bearer_cap, progress, complete, key_fac, called_num, calling_num);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating whether the port is of type NT or TE
char bearer_cap;	a character indicating the type of call, i.e. speech or data
char progress;	a character indicating the progress indicator
char complete;	a value indicating whether sending complete element should be included
char *key_fac	a pointer to an ASCII string with any keypad facility digits
char *called_num;	a pointer to an ASCII string with the called number
char *calling_num;	a pointer to an ASCII string with the calling number

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 SETUP message. A SETUP message is used to initiate a call. As the SETUP message takes a different form for NT and TE interface, the presence of some arguments will depend on the port type.

Message Sent
"DSxxbp(Kk)(calling#)/(called#)", where **xx** is the B-channel number, **b** is the bearer capability, **p** is the progress indicator, (**Kk**) is the keypad facility element if present, (**calling#**) is the directory number of the calling party and optionally the (**called#**) is the number being called. If the sending complete element is included, the '/' is replaced with a 'C'.

Note, that for an NT interface, the default directory number is used for the called party number if the called# is omitted and for a TE interface, the default directory number is used for the calling party number if the calling# is omitted.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_TYPE	invalid port type (not NT)
ILL_ARG	an invalid bearer capability value, complete value, or an invalid calling or called number was used
ILL_PROGRESS	invalid progress indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SETUP message to be sent to initiate a call. The board level command takes a different form depending on whether the port is configured as an NT (network termination) or a TE (terminal equipment). The treatment of the directory numbers is different for the two port types. The port type must be specified by the **port_type** argument. Valid values are ‘N’ for an NT interface and ‘T’ for a TE interface.

The XDS BRI Boards uses exclusive B channel allocation only. The B-channel specified for the SETUP message must not be in use at the time the message is sent.

The bearer capability indicates the type of call being made, i.e. voice or data. There are three valid choices:

A	3.1 kHz. audio (used for voice and modems)
D	unrestricted digital information, 64 kbps
S	speech (suitable for voice, but not modems)

For NT ports, or TE ports where Enbloc sending is used a progress indicator must be specified. This argument is used in interworking situations where the call is not completely ISDN. The valid values are:

C	Call is not end to end ISDN
O	Originating address is non-ISDN
N	No progress indicator

Note that if there are no interworking problems, the progress indicator should be set to ‘N’.
For TE ports, the called number is the subscriber number of the destination. The calling number element will use the default DN of the B-channel on which the call is being placed. The called number can be up to 15 digits long. If Overlap sending is used, the called number is set to a NULL string.

For NT ports, if the called number is a NULL string, the default DN of the B-channel will be used as the called number element, otherwise, the called number specified will be used. The calling number can be up to 15 digits long. If a called number is specified, the total number of digits in the called and calling number is restricted to 24. Overlap sending is rarely used, so the called number should either be specified or a default number used.

When using Enbloc sending, the Sending Complete element can be included to indicate that the entire called party number is present in the message. This is done by setting the value of complete to 0x1. If it is set to 0x0, the Sending Complete element will not be included.

Some systems may use the Keypad Facility element for supplemental services. Keypad Facility digits are included in the SETUP message my placing the in a string pointed to by key_fac. The digits must be from the set 0-9, #, and *.

Example
For an NT port**:**
xds_euro_l3_setup(1, 0x4, ‘N’, ‘S’, ‘N’, 0x1, ‘’, ‘’, “5551212”);

this will send a SETUP message for B-channel 4 on board 1 with a bearer capability of speech, no progress indicator, sending complete, no Keypad Facility digits, the default called number and calling number of 555-1212.

For a TE port using Enbloc sending:
xds_euro_l3_setup(1, 0x4, ‘T’, ‘D’, ’N’, 0, ‘’, “18005551212”, ‘’);

this will send a SETUP message for B-channel 4 on board 1 with a bearer capability of data, no progress indicator, no sending complete, no Keypad Facility digits, the default calling number, and a called number of 1-800-555-1212.

For a TE port using Overlap sending:
xds_euro_l3_setup(1, 0x4, ‘T’, ‘A’, ’N’, 0, ‘’, ‘’, ‘’);

this will send a SETUP message for B-channel 4 on board 1 with a bearer capability of 3.1 kHz audio, no progress indicator, no sending complete, no Keypad Facility digits, the default calling number, and no calling number. Subsequent digits would be sent using the function **xds_euro_l3_called_num.**

xds_euro_l3_suspend

xds_euro_l3_suspend(board_number, b_channel, call_id);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *call_id;	a pointer to an ASCII string used to identify the suspended call

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 SUSPEND message for the call currently associated with the specified B-Channel. A SUSPEND message is used to temporarily release the B-channel allocated to a call for call rearrangement. A hold can only be initiated by the user or terminal side of an interface.

Message Sent
“DHxxS(id)” where **xx** is the B-channel number, and **id** is the call_id value

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SUSPEND message to be sent for the call associated with the specified B-channel. This can only be done by the user or terminal side of the interface. If the suspend is accepted, a SUSPEND ACKnowledge message will be sent in response.

A SUSPEND REJect message will be sent in response if the hold is not accepted. This will include the cause for rejection.

A call identity string of up to 10 IA5 characters can be used to identify the call. This string is specified by the application and should be chosen so as to be unique over the user-network on which the user resides. If it is not unique, the call suspension will be rejected. This call identity string is then used when attempting to resume the call.

It should be noted that not all network switches support the SUSPEND feature. On others, it may be available by pre-subscription at the time the interface is ordered. Similarly, terminal equipment such as ISDN station sets may or may not support the feature.

Example

xds_euro_l3_suspend(1, 0x4, ‘Call #1’);	this will send a SUSPEND message for B-channel 4 on board 1 with a call identity of ‘Call #1’

xds_euro_l3_suspend_ack

xds_euro_l3_suspend_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 SUSPEND_ACKnowledge message for the call currently associated with the specified B-Channel. A SUSPEND_ACK message may be used to respond to a SUSPEND message and accept the release of the B-channel while retaining the call.

Message Sent
"DHxxA" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SUSPEND ACKnowledge message to be sent for the call associated with the specified B-channel. This should be done in response to a SUSPEND message. The B-channel will be released for use with another call. The application should retain the call identity received in the hold message as it will be used when the call is resumed.

Example

xds_euro_l3_suspend_ack(1, 0x4);	this will send a SUSPEND ACKnowledge message for B-channel 4 on board 1.

xds_euro_l3_suspend_rej

xds_euro_l3_suspend_rej(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the suspend reject

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to send a Layer 3 SUSPEND REJect message for the call currently associated with the specified B-Channel. A SUSPEND REJect message may be used to respond to a SUSPEND message and reject the release of the B-channel.

Message Sent
"DHxxRcc" where **xx** is the B-channel number and **cc** is the cause code

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SUSPEND REJect message to be sent for the call associated with the specified B-channel. This should be done in response to a SUSPEND message when the B-channel is not to be released. The cause value should indicate the reason for rejecting the hold. Typical reasons would be 0x45, "requested facility not implemented" if the application does not support call rearrangement or 0x3, "requested facility not subscribed" when the call rearrangement feature is not enabled for a specific interface. For a list of cause codes and their meaning, see the appropriate XDS Basic Rate ISDN Board Technical Manual (EuroISDN Version), Q.850 or the ETS 300 102.

Example

xds_euro_l3_suspend_rej(1, 4, 0x45);	this will send a SUSPEND REJect message for B-channel 4 on board 1 with a cause of 0x45, 'requested facility not mplemented'.

xds_euro_l3_text

xds_euro_l3_text(board_number, b_channel, mode, l3_text);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int mode;	a value between 0x0 and 0x3 indicating the text operation
char *l3_text	a pointer to a NULL terminated string of up to 27 ASCII characters

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to prepare text to be included in a Layer 3 message. Several modes are available that allow inclusion of text into a SETUP message, or provide for sending an INFOrmation message with text. Only NT (network termination) ports can send text.

Message Sent
"DIxxT" where **xx** is the B-channel number to send a single line of text
"DTxxC" where **xx** is the B-channel and the text buffer is to be cleared
"DTxxL(text)" where **xx** is the B-channel and **text** is to be put in the buffer
"DTxxA(text)" where **xx** is the B-channel and **text** is to be added to the buffer

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_MODE	invalid text mode
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to prepare text to send in a Layer 3 message. Display text is held in a buffer of up to 80 ASCII characters. Text to be sent is put in a buffer associated with a B-channel. Text can either be sent as an INFOrmation message, or as part of call control messages such as an ALERTING, a CONNECT, or a SETUP message.

The "mode" controls which text operation is carried out. The valid mode values are:

0x0	clear the buffer
0x1	send the text in the buffer as an INFOrmation message
0x2	place text in the buffer
0x3	add text to the buffer

Each B-channel has its own 80 character buffer. If a SETUP or other message is sent it will include any text in the buffer as an information element. The text buffer is then cleared. Some equipment will only accept a maximum of 32 characters. The text should be ASCII characters. Up to 27 characters may be added to the buffer per function call using modes 2 or 3. It is the responsibility of the application to format the text for readability.

Example

xds_euro_l3_text(1, 0x4, 0, 0);	this will clear the text buffer for B-channel 4 on board 1.
xds_euro_l3_text(1, 0x4, 1, 0);	this will send the text in the buffer as an INFORMATION message
xds_euro_l3_text(1, 0x4, 2, "This is line 1");	this will place 'This is line 1" in the text buffer for B channel 4 on board 1.
xds_euro_l3_text(1, 0x4, 3, "This is more text");	this will add "This is more text" to the text buffer for B-channel 4 on board 1.

xds_euro_set_bri_dn

xds_euro_set_bri_dn(board_number, b_channel, dn);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char* dn;	a pointer to a character array containing an ASCII string of up to 15 digits with the directory number

Applicable boards
All EuroISDN BRI boards

Purpose
This function is used to set the default DN or directory or subscriber number associated with a particular B-channel.

Message Sent
"SDddddddd" where **ddddddd** is the directory or subscriber number for the B-channel.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	directory number is NULL or too long
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
On a port defined as a network termination or NT, the number is used as the called party number in SETUP messages. If the port is defined as a TE, the number is the calling party in SETUP messages. Typically both B-channels on an interface will have the same directory number. If the directory number is not set with this function, it must be included in the arguments for **xds_euro_l3_setup.**

The directory number can be save in EEAROM using the **xds_set_config** function. If this is done, this and other information will automatically be restored on a board reset. Once set, this function will only be needed if the configuration changes.

Example
xds_euro_set_bri_dn(1, 0x4, "5551000");   this will set the DN for B-channel 4 to 555-1000

Basic Rate ISDN
INS-Net 64
Layer 3 ‘D’ Functions

This page was intentionally left blank.

xds_net64_l3_alert

xds_net64_l3_alert(board_number, b_channel, progress, signal, feature, indicator);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the progress indicator if any
char signal;	a character indicating the signal element if any
int feature;	a value between 0 and 63 specifying the feature indicator, if any, to option out of this feature us a (-1)
char ind;	a character indicating the state of the feature indicator

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 ALERTing message for the call currently associated with the specified B-Channel. An ALERTing message is used to notify the caller that the called party is being informed of an incoming call.

Message Sent
"DAxx(p(s(ffi))" where **xx** is the B-channel number, **p** is the optional progress indicator, **s** is the optional signal, **ff** is the optional feature indicator, and **i** is the indicator state.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid indicator value
ILL_PROGRESS	invalid progress indicator value
ILL_SIGNAL	invalid signal value
ILL_FEATURE	invalid feature value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an ALERTing message to be sent for the call associated with the specified B-channel. If the port is configured as an NT (network termination), progress indication, signal and feature indication information elements can be sent as part of the message.

Valid values for the optional progress indicator are:

C	Call not end-to-end ISDN
D	Destination not ISDN
I	Inband information or appropriate pattern now available
N	No progress indicator

For supplemental services, the signal and feature indication elements may be added, valid signal values are:

D	Dial tone
W	Call waiting tone
F	Tones off
A	Alerting tone
N	No signal

If the feature indicator value is non-zero, the feature indicator state must be set to a valid value. Valid values for the feature indication state are:

D	deactivation, indicator off
A	activation, indicator on
Q	prompt, indicator blink slow
P	pending, indicator blink fast

Example

xds_net64_l3_alert(16, 0x4, ‘I’, ‘F’, 5, ‘A’);	this will send an ALERTing message for B-channel 4 on board 16. The progress indicator is set for inband signal, signal to tones off, feature indicator 5 active.

xds_net64_l3_connect

xds_net64_l3_connect(board_number, b_channel, progress, signal, feature, indicator);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the progress indicator if any
char signal;	a character indicating the signal element if any
int feature;	a value between 0 and 63 specifying the feature indicator if any, to option out of this feature us a (-1)
char ind;	a character indicating the state of the feature indicator

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 CONNect message for the call currently associated with the specified B-Channel. A CONNect message is used to notify the caller that the called party has answered an incoming call.

Message Sent
"DCxx(p(s(ffs)))" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid indicator value
ILL_PROGRESS	invalid progress indicator value
ILL_SIGNAL	invalid signal value
ILL_FEATURE	invalid feature value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a CONNect message to be sent for the call associated with the specified B-channel. This should be done when the called party answers.

Valid values for the progress indicator are:

C	Call not end to end ISDN
D	Destination not ISDN
I	Inband information or appropriate pattern now available
O	Origination not ISDN
R	Return to ISDN
N	No progress indicator

For supplemental services, the signal and feature indication elements may be added, valid signal values are:

D	Dial tone
W	Call waiting tone
F	Tones off
A	Alerting tone
N	No signal

If the feature indicator value is non-zero, the feature indicator state must be set to a valid value. Valid values for the feature indication state are:

D	deactivation, indicator off
A	activation, indicator on
Q	prompt, indicator blink slow
P	pending, indicator blink fast

Example

xds_net64_l3_connect(1, 0x4, ‘N’, ‘N’, 0, 0);	this will send a CONNect message for B-channel 4 on board 1 with no progress indicator, signal, or feature indication.

xds_net64_l3_disconnect

xds_net64_l3_disconnect(board_number, b_channel, cause, progress, signal, feature, indicator, charge);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the disconnect
char progress;	a character indicating the progress indicator if any
char signal;	a character indicating the signal element if any
int feature;	a value between 0 and 63 specifying the feature indicator, if any, to option out of this feature us a (-1)
char ind;	a character indicating the state of the feature indicator
char *charge;	a pointer to a string of IA5 characters indicating the charge, in Yen

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 DISConnect message for the call currently associated with the specified B-Channel. A DISConnect message is used to notify either the network or the user that a disconnect sequence has been initiated.

Message Sent
"DDxxcc(p(s(ffi)($xx)" where **xx** is the B-channel number, **cc** is the cause code, **p** is the optional progress indicator, **s** is the optional signal, **ffi** is the optional feature indicator element, and **$xx** is the optional advise of charge.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_ARG	invalid indicator value
ILL_PROGRESS	invalid progress indicator value
ILL_SIGNAL	invalid signal value
ILL_FEATURE	invalid feature value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

This command is used to cause a DISConnect message to be sent for the call associated with the specified B-channel. This should be done when initiating a disconnect sequence.

The cause code is a value used to indicate the reason for a particular message, in this case, why a disconnect is occurring. A cause value of 0x10 indicates normal clearing. Other values may indicate error conditions. For a list of cause codes and their meanings, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

Valid values for the progress indicator are:

C	Call not end to end ISDN
D	Destination not ISDN
I	Inband information or appropriate pattern now available
O	Origination not ISDN
R	Return to ISDN
N	No progress indicator

For supplemental services, the signal and feature indication elements may be added, valid signal values are:

D	Dial tone
W	Call waiting tone
F	Tones off
A	Alerting tone
N	No signal

If the feature indicator value is non-zero, the feature indicator state must be set to a valid value. Valid values for the feature indication state are:

D	deactivation, indicator off
A	activation, indicator on
Q	prompt, indicator blink slow
P	pending, indicator blink fast

The optional Advise of Charge is an IA5 character string indicating the charge for the call in Yen.

Example

xds_net64_l3_disconnect(16, 0x4, 0x10, 'N', 'F', 0, 0, '');	this will send a DISConnect message for B-channel 4 on board 16, with a cause of 0x10 (normal clearing), no progress indicator, a signal of tones off, and no feature indicator or advise of charge.

xds_net64_l3_disc_ref

xds_net64_l3_disc_ref(board_number, b_channel, cause, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the disconnect
int reference;	a value between 0x00 and 0xFF that indicates the call reference for the call being released if it is not allocated a B-channel

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 DISConnect message for the call currently associated with the specified B-Channel. A DISConnect message is used to notify either the network or the user that a disconnect sequence has been initiated.

Message Sent
"DDxxccRrr" where **xx** is the B-channel number and **cc** is the cause code and **rr** is the call reference.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	invalid reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a DISConnect message to be sent for the call associated with the specified B-channel. This should be done when initiating a disconnect sequence.

The cause code is a value used to indicate the reason for a particular message, in this case, why a disconnect is occurring. A cause value of 0x10 indicates normal clearing. Other values may indicate error conditions. For a list of cause codes and their meanings, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

Example

xds_net64_l3_disconnect(1, 0x4, 0x10, 0x12);	this will send a DISConnect message for B-channel 4 on board 1, with a cause of 0x10 (normal clearing) and call reference of 0x12.

xds_net64_l3_feature_act

xds_net64_l3_feature_ind(board_number, b_channel, feature, block_id);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int feature;	a value between 0 and 63 specifying the feature activation, if any, to option out of this feature us a (-1)
char block_id;	a character between 0 and 3 identifying the blocked channel

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message with a Feature Activation element to control supplemental services.

Message Sent
"DFxxff(Bb)" where **xx** is the B-channel number, **ff** is the feature activation number, and **Bb** is the optional Blocking channel ID.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid blocking channel ID value
ILL_FEATURE	invalid feature value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a INFORMATION message with a Feature Activation element to be sent from a User port (TE). The Feature Activation element is used to control supplemental services, which are specific to the particular switch and network.

The optional Blocking Channel ID element is used to indicate that a B-channel is not available to receive incoming calls. Valid values are '0' if no channel blocks, '1' if the B1 channel is blocked, '2' if the B2 channel is blocked, and '3' if both channels block. If a 0x0, no element is included.

Example

xds_net64_l3_feature_act(16, 0x4, 5, 0);	this will send an INFORMATION message for B-channel 4 on board 16 with a feature activation code of 5 and no blocking id element.

xds_net64_l3_feature_ind

xds_net64_l3_feature_ind(board_number, b_channel, feature, indicator, block_id, charge);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int feature;	a value between 0 and 63 specifying the feature indicator, if any, to option out of this feature us a (-1)
char ind;	a character indicating the state of the feature indicator
char block_id;	a character between 0 and 3 identifying the blocked channel
char *charge;	a pointer to a string of IA5 characters indicating the charge, in Yen

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message with a Feature Indication element to control supplemental services. An optional blocking channel ID or advise of charge element may be included.

Message Sent
“DFxxffi(Bb)” or “DFxxffi($xx)” where **xx** is the B-channel number, **ffi** is the feature indication and indictor status, (**Bb**) is the optional blocking channel id and (**$xx**) is the advise of charge string.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid indicator, or blocking channel ID value
ILL_FEATURE	invalid feature value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a INFORMATION message with a Feature Indication element to be sent. The Feature Indication element is used to control supplemental services which are specific to the particular switch and network. If the feature indicator value is non-zero, the feature indicator state must be set to a valid value. Valid values for the feature indication state are:

D	deactivation, indicator off
A	activation, indicator on
Q	prompt, indicator blink slow
P	pending, indicator blink fast

Either the Blocking Channel ID or Advise of Charge element may optionally be included in the message. The optional Blocking Channel ID element is used to indicate that a B-channel is not available to receive incoming calls. Valid values are ‘0’ if no channel blocks, ‘1’ if the B1 channel is blocked, ‘2’ if the B2 channel is blocked, and ‘3’ if both channels block. If a 0x0, no element is included. The Advise of Charge is used to indicate an amount in Yen that is the charge for the previous call.

Example

xds_net64_l3_feature_ind(16, 0x4, 5, ‘P’, 0, “$237”);	this will send an INFORMATION message for B-channel 4 on board 16 with a feature indication element 5 set to prompt, and an advise of charge of 237 Yen.

xds_net64_l3_hold

xds_net64_l3_hold(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 HOLD message to a B-channel.

Message Sent
"DHxx" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD message to a selected b-channel on a selected board.

Example

xds_net64_l3_hold(1, 0x4);	this sends a message (B-channel 4 to board 1)

xds_net64_l3_hold_ack

xds_net64_l3_hold_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 HOLD ACKnowledge message to a B-channel.

Message Sent
"DHxxA" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD ACKnowledge message to a selected b-channel on a selected board.

Example

xds_net64_l3_hold_ack(1, 0x4);	this sends a message (B-channel 4 to board 1)

xds_net64_l3_hold_rej

xds_net64_l3_hold_rej(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the hold rejection

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 HOLD REJect message to a B-channel.

Message Sent
"DHxxRcc" where **xx** is the B-channel number, and cc is the cause for the rejection

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD REJect message to a selected b-channel on a selected board.

Example

xds_net64_l3_hold_rej(1, 0x4, 0x7F);	this sends a message (B-channel 4 to board 1, cause 0x7F)

xds_net64_l3_info

xds_net64_l3_info(board_number, b_channel, cause, key_fac, feature, indicator, block_id, charge);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the disconnect
char *key_fac	a pointer to an ASCII string with any keypad facility digits
int feature;	a value between 0 and 63 specifying the feature indicator, if any, to option out of this feature us a (-1)
char ind;	a character indicating the state of the feature indicator
char block_id;	a character between 0 and 3 identifying the blocked channel
char *charge;	a pointer to a string of IA5 characters indicating the charge, in Yen

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message from the network with optional cause, Keypad Facility, feature indication, blocking channel id or advise of charge elements.

Message Sent
"DIxx(cc)(Kkk)(ffi)(Bb)($xx)" where **xx** is the B-channel number, **cc** is the optional cause, **Kkk** are the optional Keypad digit(s), **ffi** is the optional feature indication, and **Bb** or **$xx** are the optional blocking channel id or advise of charge elements.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid indicator value
ILL_FEATURE	invalid feature value
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFORMATION message to be sent from a port set to the network type. An optional cause element may be included if the cause value is non-zero. The Keypad Facilities or feature indication elements may be included to control supplemental services, which are specific to the particular switch and network. The Keypad Facilities element consists of one or more keypad digits as found on a telephone set keypad, typically in the range 0-9, *, and #.

If the feature indicator value is non-zero, the feature indicator state must be set to a valid value. Valid values for the feature indication state are:

D	deactivation, indicator off
A	activation, indicator on
Q	prompt, indicator blink slow
P	pending, indicator blink fast

Either the Blocking Channel ID or Advise of Charge element may optionally be included in the message. The optional Blocking Channel ID element is used to indicate that a B-channel is not available to receive incoming calls. Valid values are ‘0’ if no channel blocks, ‘1’ if the B1 channel is blocked, ‘2’ if the B2 channel is blocked, and ‘3’ if both channels block. If a 0x0, no element is included. The Advise of Charge is used to indicate an amount in Yen that is the charge for the previous call.

Example

xds_net64_l3_info(1, 0x4, 0, ‘2*’, 0, 0, 0, ‘’);	this will send an INFORMATION message for B-channel 4 on board 1 with Keypad Facility string of ‘2*’ and no cause, feature indication, blocking channel id or advise of charge elements

xds_net64_l3_info_text

xds_net64_l3_info_text(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message with the display text continued in the buffer for that channel.

Message Sent
“DIxxT” where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a INFORMATION message with a Display element consisting of the text in the buffer for the specified channel to be sent. Text may be placed in the buffer using the **xds_net64_l3_text** function.

Example

xds_net64_l3_info_text(1, 0x4);	this will send an INFORMATION message for B-channel 4 on board 16 with a Display element consisting of the text in the buffer.

xds_net64_l3_notify

xds_net64_l3_notify(board_number, b_channel, notification);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char notification;	a character indicating the notification type

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 NOTIFY message to inform a user or terminal of the change of state of a call due to call rearrangement.

Message Sent
"DNxxn" where **xx** is the B-channel number and **n** is the notification indicator.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid notification indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a NOTIFY message to be sent from an NT port to a user or terminal. The notification is used to inform the terminal of a change of state of a call due to call rearrangement so that the terminal may reflect the change of state by changing the state of an indicator (i.e. controlling whether an indicator blinks or not) or other means of informing the user.

The valid notification indicators are:

R - Call Resumed
S - Call Suspended

Example

xds_net64_l3_notify(1, 0x4, 'R');	this will send a NOTIFY message for B-channel 4 on board 1 with a notification of Call Resumed.

xds_net64_l3_proceed

xds_net64_l3_proceed(board_number, b_channel, progress, signal, feature, indicator);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the progress indicator if any
char signal;	a character indicating the signal element if any
int feature;	a value between 0 and 63 specifying the feature indicator, if any, to option out of this feature us a (-1)
char ind;	a character indicating the state of the feature indicator

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 CALL PROCeeding message for the call currently associated with the specified B-Channel. A CALL PROCeeding message is used to notify the caller that all information necessary to process a call has been received and that call establishment has been initiated.

Message Sent
"DPxx(p(s(ffi))" where **xx** is the B-channel number, **p** is the progress indicator, **s** is the optional signal, and **ffi** is the optional feature indication.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid indicator value
ILL_FEATURE	invalid signal indicator value
ILL_SIGNAL	invalid progress indicator value
ILL_PROGRESS	invalid progress indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a CALL PROCeeding message to be sent for the call associated with the specified B-channel. This should be done in response to a SETUP message for Enbloc sending, or when all necessary digits have been received for Overlap sending when the call can be initiated.

The progress indicator may take values of:

C	Call not end to end ISDN
D	Destination not ISDN
I	Inband information or appropriate pattern now available
O	Origination not ISDN
R	Return to ISDN
N	No progress indicator

For supplemental services, the signal and feature indication elements may be added, valid signal values are:

D	Dial tone
W	Call waiting tone
F	Tones off
A	Alerting tone
N	No signal

If the feature indicator value is non-zero, the feature indicator state must be set to a valid value. Valid values for the feature indication state are:

D	deactivation, indicator off
A	activation, indicator on
Q	prompt, indicator blink slow
P	pending, indicator blink fast

Example

xds_net64_l3_proceed(16, 0x4, ‘D’, ‘F’, 0, 0);	this will send a CALL PROCeeding message for B-channel 4 on board 1, with a progress indicator of destination not end ISDN, signal of tones off, and no feature indication.

xds_net64_l3_progress

xds_net64_l3_progress(board_number, b_channel, cause, progress, signal);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x0 and 0xff that indicates the reason for the PROGress message
char progress;	a character indicating the progress indicator, if any
char signal;	a character indicating the signal element, if any

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 PROGress message for the call currently associated with the specified B-Channel. A PROGress message is used to notify the caller of interworking with non-ISDNs, a call is routed to an inband tone or announcement, or when a progress delay at the destination is reported.

Message Sent
"DPxxPccp(s)" where **xx** is the B-channel number, **cc** is the cause value, **p** is the progress indicator, and **s** is the optional signal

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_SIGNAL	invalid progress indicator value
ILL_PROGRESS	invalid progress indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a PROGress message to be sent for the call associated with the specified B-channel. A PROGress message is sent to report an interworking situation such as the destination is non ISDN, or a situation where the call, though initiated can not be completed, such as the called party is busy. In those cases, an inband signal such as busy tone is usually present and indicated by the progress indicator and signal elements in the message. An application would send a PROGress message when it realizes that a call can not be completed, or when interworking with the destination switch results in the receipt of a PROGress message from the destination.

The cause code is a value used to indicate the reason for the PROGress message. Examples would be a cause code of 0x11, “User busy”, or 0x12, “No user responding”. For a list of cause codes and their meaning, see the XDS Layer 3 Protocol Software Reference Manual or Q.850. Progress values that are valid may be:

C	Call is not end to end ISDN
D	Destination is non-ISDN
I	Inband information or appropriate pattern now available
O	Origination is non-ISDN
R	Return to ISDN
N	no progress indicator

For supplemental services, the signal and feature indication elements may be added, valid signal values are:

D	Dial tone
W	Call waiting tone
F	Tones off
A	Alerting tone
N	No signal

Example

xds_net64_l3_progress(1, 0x4, 0x11, ‘I’, ‘N’);	this will send a PROGress message for B-channel 4 on board 1. The cause is 0x11, “User busy”, the progress indicator is inband information, and no signal.

xds_net64_l3_query_status

xds_net64_l3_query_status(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 STATus Enquiry message to the board. This can be done to prompt a terminal to respond with a STATUS message.

Message Sent
"DXxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to query an ISDN terminal about the call state of a call in a case where a received message indicates there may be a call processing discrepancy. The terminal, if it responds will send a STATUS message. As this response is dependent on the terminal and not on the board, this message will be returned on the receive message queue and will be retrieved with the **xds_msg_receive** function rather than the **xds_query_receive** function. This function is only valid for B-channels that have a call in process.

Example

xds_net64_l3_query_status(0x1, 0x4)	This would send a STATus ENQuiry message for the B1 channel of the third port on board 1.

xds_net64_l3_rel_com

xds_net64_l3_rel_com(board_number, b_channel, cause, signal, feature, indicator, charge);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the release
char signal;	a character indicating the signal element, if any
int feature;	a value between 0 and 63 specifying the feature indicator, if any, to option out of this feature us a (-1)
char ind;	a character indicating the state of the feature indicator
char charge;	a pointer to a string of IA5 characters indicating the charge, in Yen

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 RELease or REL_COMplete message for the call currently associated with the specified B-Channel. A RELease or REL_COMplete message is used to release a call reference and terminate a call in cases where a DISConnect message is inappropriate.

Message Sent
"DRxxcc(s(ffs)($xx)" where **xx** is the B-channel number, **cc** is the cause value, **ff** is the feature number, and **rr** is the optional call reference.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_SIGNAL	invalid progress indicator value
ILL_FEATURE	invalid feature value
ILL_ARG	invalid indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a RELease REL_COMplete message to be sent for the call associated with the specified B-channel. The message type will depend on the call state of the call. This should be done to release a call and associated call reference under some circumstances. An example is when no TE equipment has responded to a SETUP message. If the call has been allocated a B-channel, a DISConnect should be sent to clear a call (see **xds_net64_l3_disconnect**).

The cause code is a value used to indicate the reason for the REL_COMplete message. Examples would be a cause code of 0x10, “Normal clearing”, or 0x41, “Bearer capability not supported”. For a list of cause codes and their meaning, see the XDS Layer 3 Protocol Software Reference Manual or Q.850. The signal element is used to indicate the presence or absence of inband signals and should be ‘F’ for tones of or ‘N’. The Feature Indication element is used for supplemental services. The advise of charge element consists of a string of IA5 characters indicating the call charge in Yen.

Example

xds_net64_l3_rel_com(16, 0x4, 0x10, ‘N’, 0, ‘’);	this will send a RELease COMplete message for B-channel 4 on board 16 with a cause value of 0x10, “Normal Clearing” and no signal, feature indication or advise of charge element.

xds_net64_l3_rel_com_ref

xds_net64_l3_rel_com_ref(board_number, b_channel, cause, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the release
int reference;	a value between 0x00 and 0xFF that indicates the call reference for the call being released

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 REL_COMplete message for the call specified by the call reference. A REL_COMplete message is used to release a call reference and terminate a call in cases where a DISConnect message is inappropriate.

Message Sent
"DRxxccRrr" where **xx** is the B-channel number **cc** is the cause value and **rr** is the call reference.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	invalid call reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a REL_COMplete message to be sent for the call associated with the specified B-channel. This should be done to release a call and associated call reference under some circumstances. An example is when no TE equipment has responded to a SETUP message. If the call has been allocated a B-channel, a DISConnect should be sent to clear a call (see **xds_net64_l3_disconnect**).

The cause code is a value used to indicate the reason for the message. Examples would be a cause code of 0x10, “Normal clearing”, or 0x41, “Bearer capability not supported”. For a list of cause codes and their meaning, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

Example

xds_net64_l3_rel_com_ref(16, 0x4, 0x10, 0x12);	this will send a RELease COMplete message for B-channel 4 on board 16 with a cause value of 0x10, “Normal Clearing” and a call reference of 0x12.

xds_net64_l3_resume

xds_net64_l3_resume(board_number, b_channel, call_id);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *call_id;	a pointer to an IA5 string used to identify the suspended call

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 RESUME message for the call specified by the call identity. A RESUME message is used to reestablish a connection to a suspended call.

Message Sent
"DGxxS(id)" where **xx** is the B-channel number and id is the call identity string.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid calling number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a RESUME message to be sent for the call with the specified call identity. The call will be allocated the specified B-channel. This is done when resuming a call previously suspended using a SUSPEND message. The selected B-channel must be available. The NT side should respond with either a RESUME ACKnowledge message, in which case, the connection is restored, or a RESUME REJect message, in which case, the call remains in the suspended state.

The call identity used in this message is the one used in the original SUSPEND message. It is the responsibility of the application to retain this call identity as long as the call is in the suspended state. Failure to do so may result in unretrievable or unreleasable calls.

Example

xds_net64_l3_resume(16, 0x4, "Call #1");	this will send a RESUME message for B-channel 4 on board 16 with a call identity of "Call #1"

xds_net64_l3_resume_ack

xds_net64_l3_resume_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 RESUME ACKnowledge message in response to a RESUME message. The call will be allocated the specified B-channel.

Message Sent
"DGxxA" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a RESUME ACKnowledge message to be sent in response to a RESUME message. The call will be allocated the specified B-channel. Note, that this channel must match that received in the RESUME message. If a RESUME ACKnowledge message is sent, the connection to the call is reestablished on the specified B channel. The Call Identity of the resumed call is then free. Note, that as only a TE (terminal equipment) port can send a RESUME message, only an NT (network termination) port can grant the resumption by sending a RESUME ACKnowledge message.

Example

xds_net64_l3_resume_ack(16, 0x4);	this will send a RESUME ACKnowledge message for B-channel 4 on board 16.

xds_net64_l3_resume_rej

xds_net64_l3_resume_rej(board_number, b_channel, cause, charge);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for rejecting the resumption
char *charge;	a pointer to a string of IA5 characters indicating the charge, in Yen

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 RESUME REJect message for the call on the specified B-channel. A RESUME REJect message is used to deny a RESUME request message.

Message Sent
"DGxxRcc($xx)" where **xx** is the B-channel number and **cc** is the cause, and **$xx** is the optional advise for charge.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a RESUME REJect message to be sent for the call on the specified B-channel. The B-channel should match that in the received RESUME message. Note, that as only a TE (terminal equipment) port can send a RESUME message, and only an NT (network termination) port can grant the retrieval by sending a RESUME REJect message.
The cause code is a value used to indicate the reason for the PROGress message. Examples would be a cause code of 0x06, “Channel unacceptable”, or 0x2A, “Switching equipment congestion”. For a list of cause codes and their meaning, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

An optional advise of charge element can be included which gives the charge for the call in yen.

Example

xds_net64_l3_resume_rej(16, 0x4, 0x2A);	this will send a RESUME REJect message for B-channel 4 on board 16, with a cause of 0x2A, “Switching equipment congestion”.

xds_net64_l3_retrieve

xds_net64_l3_retrieve(board_number, b_channel, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int reference;	a value between 0x0 and 0xFF that indicates the reference number of the call

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 RETrieve message to a B-channel.

Message Sent
"DGxxrr" where **xx** is the B-channel number, and **rr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	invalid call reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a RETrieve message to a selected b-channel on a selected board.

Example

xds_net64_l3_retrieve(1, 0x4, 0x93);	this sends a message (B-channel 4 to board 1 with a call reference # of 0x93)

xds_net64_l3_retrieve_ack

xds_net64_l3_retrieve_ack(board_number, b_channel, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int reference;	a value between 0x0 and 0xFF that indicates the reference number of the call

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 RETrieve ACKnowledge message to a B-channel.

Message Sent
"DGxxArr" where **xx** is the B-channel number, and **rr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	invalid call reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a RETrieve ACKnowledge message to a selected b-channel on a selected board.

Example

xds_net64_l3_retrieve_ack(1, 0x4, 0x93);	this sends a message (B-channel 4 to board 1 with a call reference # of 0x93)

xds_net64_l3_retrieve_rej

xds_net64_l3_retrieve_rej(board_number, b_channel, cause, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x0 and 0xFF that indicates the cause value of the rejection
int reference;	a value between 0x0 and 0xFF that indicates the reference number of the call

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 RETrieve REJect message to a B-channel.

Message Sent
"DGxxArr" where **xx** is the B-channel number, and **rr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	invalid call reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a RETrieve REJect message to a selected b-channel on a selected board.

Example

xds_net64_l3_retrieve_ack(1, 0x4, 0x19, 0x93);	this sends a message (B-channel 4 to board 1 with a cause value of 0x19 and a call reference # of 0x93)

xds_net64_l3_setup

xds_net64_l3_setup(board_number, b_channel, port_type, bearer_cap, progress, key_fac, called_num, calling_num);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating whether the port is of type NT or TE
char bearer_cap;	a character indicating the type of call, i.e. speech or data
char progress;	a character indicating the progress indicator
char *key_fac	a pointer to an ASCII string with any keypad facility digits
char *called_num;	a pointer to an ASCII string with the called number
char *calling_num;	a pointer to an ASCII string with the calling number

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 SETUP message. A SETUP message is used to initiate a call. As the SETUP message takes a different form for NT and TE interface, the presence of some arguments will depend on the port type.

Message Sent
"DSxxbp(Kk)(calling#)/(called#)", where **xx** is the B-channel number, **b** is the bearer capability, **p** is the progress indicator, (**Kk**) is the keypad facility element if present, (**calling#**) is the directory number of the calling party and optionally the (**called#**) is the number being called. Note, that for an NT interface, the default directory number is used for the called party number if the **called#** is omitted and for a TE interface, the default directory number is used for the calling party number if the **calling#** is omitted.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
ILL_TYPE	invalid port type
ILL_ARG	invalid bearer capability value, calling number, or called number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SETUP message to be sent to initiate a call. The board level command takes a different form depending on whether the port is configured as an NT (network termination) or a TE (terminal equipment). The treatment of the directory numbers is different for the two port types. The port type must be specified by the **port_type** argument. Valid values are ‘N’ for an NT interface and ‘T’ for a TE interface.

The XDS BRI Boards uses exclusive B channel allocation only. The B-channel specified for the SETUP message must not be in use at the time the message is sent.
The bearer capability indicates the type of call being made, i.e. voice or data. There are three valid choices:

A	3.1 kHz. audio (used for voice and modems)
D	unrestricted digital information, 64 kbps
S	speech (suitable for voice, but not modems)

For NT ports, or TE ports where Enbloc sending is used a progress indicator must be specified. This argument is used in interworking situations where the call is not completely ISDN. The valid values are:

C	Call is not end to end ISDN
O	Originating address is non-ISDN
N	No progress indicator

Note that if there are no interworking problems, the progress indicator should be set to ‘N’.
For TE ports, the called number is the subscriber number of the destination. The calling number element will use the default DN of the B-channel on which the call is being placed. The called number can be up to 15 digits long.

For NT ports, if the called number is a NULL string, the default DN of the B-channel will be used as the called number element, otherwise, the called number specified will be used. The calling number can be up to 15 digits long. If a called number is specified, the total number of digits in the called and calling number is restricted to 24.

Note that in INS-Net 64, overlap sending is not used, so the called number should either be specified or a default number used.

Some systems may use the Keypad Facility element for supplemental services. Keypad Facility digits are included in the SETUP message my placing the in a string pointed to by key_fac. The digits must be from the set 0-9, #, and *.

Example
For an NT port**:**
xds_net64_l3_setup(1, 0x4, ‘N’, ‘S’, ‘N’, ‘’, ‘’, “5551212”);
this will send a SETUP message for B-channel 4 on board 1 with a bearer capability of speech, no progress indicator, sending complete, no Keypad Facility digits, the default called number and calling number of 555-1212.

For a TE port using Enbloc sending:
xds_net64_l3_setup(1, 0x4, ‘T’, ‘D’, ‘N’, ‘’, “18005551212” , ‘’);

this will send a SETUP message for B-channel 4 on board 1 with a bearer capability of data, no progress indicator, no Keypad Facility digits, the default calling number, and a called number of 1-800-555-1212.

xds_net64_l3_suspend

xds_net64_l3_suspend(board_number, b_channel, call_id);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *call_id;	a pointer to an ASCII string used to identify the suspended call

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 SUSPEND message for the call currently associated with the specified B-Channel. A SUSPEND message is used to temporarily release the B-channel allocated to a call for call rearrangement. A hold can only be initiated by the user or terminal side of an interface.

Message Sent
"DHxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SUSPEND message to be sent for the call associated with the specified B-channel. This can only be done by the user or terminal side of the interface. If the suspend is accepted, a SUSPEND ACKnowledge message will be sent in response.
A SUSPEND REJect message will be sent in response if the hold is not accepted. This will include the cause for rejection.

A call identity string of up to 10 IA5 characters can be used to identify the call. This string is specified by the application and should be chosen so as to be unique over the user-network on which the user resides. If it is not unique, the call suspension will be rejected. This call identity string is then used when attempting to resume the call.

It should be noted that not all network switches support the SUSPEND feature. On others, it may be available by presubscription at the time the interface is ordered. Similarly, terminal equipment such as ISDN station sets may or may not support the feature.

Example

xds_ins64_l3_suspend(16, 0x4, “Call #1”);	this will send a SUSPEND message for B-channel 4 on board 16 with a call identity of “Call #1”.

xds_net64_l3_suspend_ack

xds_net64_l3_suspend_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 SUSPEND_ACKnowledge message for the call currently associated with the specified B-Channel. A SUSPEND_ACK message may be used to respond to a SUSPEND message and accept the release of the B-channel while retaining the call.

Message Sent
"DHxxA" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SUSPEND ACKnowledge message to be sent for the call associated with the specified B-channel. This should be done in response to a SUSPEND message. The B-channel will be released for use with another call. The application should retain the call identity received in the hold message as it will be used when the call is resumed.

Example

xds_net64_l3_suspend_ack(16, 0x4);	this will send a SUSPEND ACKnowledge message for B-channel 4 on board 16.

xds_net64_l3_suspend_rej

xds_net64_l3_suspend_rej(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the suspend reject

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 SUSPEND REJect message for the call currently associated with the specified B-Channel. A SUSPEND REJect message may be used to respond to a SUSPEND message and reject the release of the B-channel.

Message Sent
"DHxxRcc" where **xx** is the B-channel number and cc is the cause code

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SUSPEND REJect message to be sent for the call associated with the specified B-channel. This should be done in response to a SUSPEND message when the B-channel is not to be released. The cause value should indicate the reason for rejecting the hold. Typical reasons would be 0x45, "requested facility not implemented" if the application does not support call rearrangement or 0x3, "requested facility not subscribed" when the call rearrangement feature is not enabled for a specific interface. For a list of cause codes and their meaning, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

Example

xds_net64_l3_suspend_rej(16, 4, 0x45);	this will send a SUSPEND REJect message for B-channel 4 on board 16 with a cause of 0x45.

xds_net64_l3_text

xds_net64_l3_text(board_number, b_channel, mode, l3_text);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int mode;	a value between 0x0 and 0x3 indicating the text operation
char *l3_text;	a pointer to a NULL terminated string of up to 27 IA5 characters

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to prepare text to be included in a Layer 3 message. Several modes are available that allow inclusion of text into a SETUP message, or provide for sending an INFOrmation message with text. Only NT (network termination) ports can send text.

Message Sent
"DIxxT" where **xx** is the B-channel number to send a single line of text
"DTxxC" where **xx** is the B-channel and the text buffer is to be cleared
"DTxxL(text)" where **xx** is the B-channel and **text** is to be put in the buffer
"DTxxA(text)" where **xx** is the B-channel and **text** is to be added to the buffer

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_MODE	invalid text mode
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to prepare text to send in a Layer 3 message. Display text is held in a buffer of up to 32 ASCII characters. Text to be sent is put in a buffer associated with a B-channel. Text can either be sent as an INFOrmation message, or as part of call control messages such as an ALERTING, a CONNECT, or a SETUP message.

The "mode" controls which text operation is carried out. The valid mode values are:

0x0	clear the buffer
0x1	send the text in the buffer as an INFOrmation message
0x2	place text in the buffer
0x3	add text to the buffer

Each B-channel has its own 32 character buffer. If a SETUP or other message is sent it will include any text in the buffer as an information element. The text buffer is then cleared. The text should be ASCII characters. Up to 27 characters may be added to the buffer per function call using modes 2 or 3. It is the responsibility of the application to format the text for readability.

Example

xds_net64_l3_text(16, 0x4, 0, 0);	this will clear the text buffer for B-channel 4 on board 16.
xds_net64_l3_text(16, 0x4, 1, 0);	this will send the text in the buffer as an INFORMATION message
xds_net64_l3_text(16, 0x4, 2, "This is line 1");	this will place "This is line 1" in the text buffer for B channel 4 on board 16.
xds_net64_l3_text(16, 0x4, 3, "This is more text");	this will add "This is more text" to the text buffer for B-channel 4 on board 16.

xds_net64_l3_user_alert

xds_net64_l3_user_alert(board_number, b_channel, progress, feature);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the progress indicator if any
int feature;	a value between 0 and 63 specifying the feature activation if any, to option out of this feature us a (-1)

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 ALERTing message for the call currently associated with the specified B-Channel. An ALERTing message is used to notify the caller that the called party is being informed of an incoming call.

Message Sent
"DAxxpff" where **xx** is the B-channel number, **p** is the optional progress indicator and **ff** is the optional feature activation.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator mode
ILL_FEATURE	invalid feature indicator mode
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an ALERTing message to be sent from a user port (TE) for the call associated with the specified B-channel. Progress indication and feature activation information elements can be sent as part of the message.

Valid values for the optional progress indicator are:

C	Call not end-to-end ISDN
D	Destination not ISDN
N	No progress indicator

For supplemental services the feature activation element may be added.

Example

xds_net64_l3_user_alert(16, 0x4, ‘D’, 5,);	this will send an ALERTing message for B-channel 4 on board 16. The progress indicator is set for destination not ISDN, feature activation 5.

xds_net64_l3_user_info

xds_net64_l3_user_info(board_number, b_channel, cause, key_fac, feature, block_id);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the disconnect
char *key_fac	a pointer to an ASCII string with any keypad facility digits
int feature;	a value between 0 and 63 specifying the feature activation, if any, to option out of this feature us a (-1)
char block_id;	a character between 0 and 3 identifying the blocked channel

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 INFOrmation message from a user (terminal). Optional cause, Keypad Facility, Feature, of Blocking Channel ID elements may be included.

Message Sent
"DIxx(cc)(Kkk)(ff)(Bb)" where **xx** is the B-channel number, **cc** is the optional cause indicator, **Kkk** is the optional Keypad Facility digits, **ff** is the optional feature activation, and **Bb** is the optional blocking channel id.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause mode
ILL_ARG	invalid blocking ID value
ILL_FEATURE	invalid feature indicator mode
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFOrmation message to be sent from a user port (TE) for the call associated with the specified B-channel. Optional Cause and Blocking channel ID elements may be included. Keypad Facility and Feature activation information elements may also be sent as part of the message to control supplemental services.

The cause code is a value used to indicate the reason for the message. Examples would be a cause code of 0x10, “Normal clearing”, or 0x41, “Bearer capability not supported”. For a list of cause codes and their meaning, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

The optional Blocking Channel ID element is used to indicate that a B-channel is not available to receive incoming calls. Valid values are ‘0’ if no channel blocks, ‘1’ if the B1 channel is blocked, ‘2’ if the B2 channel is blocked, and ‘3’ if both channels block. If a 0x0, no element is included

Example

xds_net64_l3_user_info(16, 0x4, 0, ‘2’, 0, 0);	this will send an INFOrmation message for B-channel 4 on board 16. A Keypad Facility digit of ‘2’ is included. No cause, feature, or blocking channel id is included.

xds_net64_l3_user_proceed

xds_net64_l3_user_proceed(board_number, b_channel, progress, feature);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the progress indicator, if any
int feature;	a value between 0 and 63 specifying the feature activation, if any, to option out of this feature us a (-1)

Applicable boards
XDS INS-Net 64 BRI boards

Purpose
This function is used to send a Layer 3 CALL PROCeeding message for the call currently associated with the specified B-Channel. A CALL PROCeeding message is used to notify the caller that all information necessary to process a call has been received and that call establishment has been initiated.

Message Sent
"DPxxp(ff)" where **xx** is the B-channel number, **p** is the progress indicator, and **ff** is the optional feature activation.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress mode
ILL_FEATURE	invalid feature indicator mode
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a CALL PROCeeding message to be sent from a user port (TE) for the call associated with the specified B-channel. This should be done in response to a SETUP message for Enbloc sending, or when an interworking situation exists.
The progress indicator may take values of:

C	Call not end to end ISDN
D	Destination not ISDN
O	Origination not ISDN
N	No progress indicator

Example

xds_net64_l3_user_proceed(1, 0x4, ‘D’, 0);	this will send a CALL PROCeeding message for B-channel 4 on board 1, with a progress indicator of destination not ISDN, and no feature activation.

This page was intentionally left blank.

Basic Rate ISDN
AT&T Custom
Layer 3 ‘D’ Functions

This page was intentionally left blank.

xds_att_l3_assoc

int xds_att_l3_assoc(board_number, b_channel, reference, type, call_app)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int reference;	a value between 0x0 and 0x1F indicating the B-channel
char type;	the call type, valid characters are ‘S’, ‘C’, ‘H’, ‘R’, ‘E’, ‘D’, ‘X’, or ‘S’
int call_app;	a value between 0 and 254 the Call Appearance number

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
The purpose of this function is to send an ASSOCiated type message. This message is used by the network to indicate to a user that activity is taking place on an associated user endpoint. It is only valid when the port is acting as the network.

Message Sent
“DBxxGrrt”, where **xx** is the b-channel, **rr** is the reference number, **t** is the type. If the call appearance number is not 0, that number will be appended to the end of the message.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_TYPE	an invalid association type was selected
ILL_ARG	an invalid call appearance was called
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
In the AT&T Custom Protocol, several users can be associated in a group that share common call appearances. When a call is presented to a group it appears at that call appearance for each member of the group. When an activity takes place such as one of the members of the group answering the call, the other users of the group are informed of this by an ASSOCiated message. The associated type element in the message indicates which type of activity has taken place. The state of the call will be indicated by the LEDs for that particular call appearance.

The first message to establish an associated call will be an ASSOCiated message with an associated type of Setup. A terminal will respond with an ASSOCiated ACKnowledge message (see **xds_att_l3_assoc_ack**). When sending an ASSOCiated message with an associated type of SETUP, the application should always use a call reference of 0. The ASSOCiated ACKnowledge message returned by the terminal will contain that call reference that should be used in all subsequent messages. For more information on the use of ASSOCiated messages see *The XDS Layer 3 Protocol Manual 251M031.*

Example

xds_att_l3_assoc(16, 0x11, 0x19, 'S', 3);	this will send and ASSOCiated type message to board 16, b-channel 11, with a call reference number of 19, a call type 'S', and a call appearance of 3

xds_att_l3_assoc_ack

int xds_att_l3_assoc_ack(board_number, b_channel, reference)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int reference;	a value between 0x0 and 0x1F indicating the B-channel

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
The purpose of this function is to send an ASSOCiated ACKnowledge type message. This message is used by the network to ACKnowledge to a user that activity is taking place on an associated user endpoint. It is only valid when the port is acting as the network. The ASSOCiated ACKnowledge message returned by the terminal will contain that call reference that should be used in all subsequent messages.

Message Sent
"DBxxGrrA", where **xx** is the b-channel, **rr** is the reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The first message to establish an associated call will be an ASSOCiated message with an associated type of Setup. The ASSOCiated ACKnowledge message returned by the terminal will contain that call reference that should be used in all subsequent messages. For more information on the use of ASSOCiated messages see *The XDS Layer 3 Protocol Manual 251M031*.

Example

xds_att_l3_assoc_ack(16, 0x12, 0x18);	this will send and ASSOCiated type message to board 16, b-channel 12, with a call reference number of 18

xds_att_l3_call_appearance

int xds_att_l3_call_appearance(board_number, b_channel, call_app, adj_ctnl)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int call_app;	a value indicating the call appearance
int adj_ctnl;	a value of ‘N’, ‘F’, or 0

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to select the active call appearance on a terminal using the AT&T Custom Protocol. It is only valid when the port is acting as the network.

Message Sent
“DBxxA=caa”, where **xx** is the b-channel, **ca** is the call_app, and the **a** is the adj_ctnl.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid adjunct control value was called
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
ISDN Station sets using the AT&T Custom Protocol have a number of call appearances which correspond to button/indicator pairs on the set. The active call appearance is indicated by an LED associated with that call appearance. Some sets have one LED which indicates both the selected call appearance and the state of the call associated with that appearance while others have two LEDs, one for selected call appearance and the other for call state. Typically, the LED will be red when the call appearance is selected.

If the user wishes to change the call appearance, an AT&T Feature Key message will be sent with the feature key corresponding to the call appearance number. The network, if it accepts the change will send an INFOrmation message with the new selected call appearance. This will change the selected call appearance LED. Any activity on the station set such as going off-hook, hanging up, or placing a call on hold will now be for the call associated with the selected call appearance.

The adjunct control element may be used to direct the station set to go on or off hook. As an example, when responding to an AT&T TRANSFER message from a user, the network may send an AT&T HOLD message followed by an INFO message with a new call appearance number (using a vacant call appearance) and the adjunct control set to off-hook. The station set will automatically respond by sending a SETUP message using the new call appearance number. The adjunct control element is optional, and can be “F” for off-hook, “N” for on-hook, or NUL if it is not to be included.

Example

xds_bri_l3_call_appearance(16, 0x12, 2, ‘F’);	this will display the call appearance for board 16, b-channel 12, with a call appearance number of 2, as being “OFF-HOOK”

xds_att_l3_conference

int xds_att_l3_conference(board_number, b_channel)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
The purpose of this function is to send a conference message to the specified b-channel.

Message Sent
"DBxxC", where **xx** is the b-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
For more information on the use of the conference messages see *The XDS Layer 3 Protocol Manual 251M031*.

Example

xds_att_l3_conference(16, 0x11);	this will send a layer 3 conference message to board 16, b-channel 11

xds_att_l3_conference_ack

int xds_att_l3_conference_ack(board_number, b_channel)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
The purpose of this function is to send a conference acknowledge message to the specified b-channel.

Message Sent
"DBxxCA", where **xx** is the b-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
For more information on the use of conference messages see *The XDS Layer 3 Protocol Manual 251M031.*

Example

xds_att_l3_conference_ack(16, 0x11);	this will send a layer 3 conference acknowledge message to board 16, b-channel 11

xds_att_l3_conference_rjc

int xds_att_l3_conference_rjc(board_number, b_channel, cause)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0 and 0xFF of the cause

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
The purpose of this function is to send a conference reject message to the specified b-channel.

Message Sent
“DBxxCRcc”, where **xx** is the b-channel, and **cc** is the cause value

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_CAUSE	an invalid rejection cause value was used
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
For more information on the use of conference messages see *The XDS Layer 3 Protocol Manual 251M031.*

Example

xds_att_l3_conference_rjc(16, 0x11, 0xFF);	this will send a layer 3 conference reject message to board 16, b-channel 11, with a cause of 0xFF

xds_att_l3_drop

int xds_att_l3_drop(board_number, b_channel)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send an AT&T DROP message. This message is used by a user to request that the last call added to a conference be removed. For a two party call, the DROP message requests that the call be disconnected. This function is only valid when the port is acting as a terminal or user.

Message Sent
"DBxxD", where **xx** is the b-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The AT&T DROP message is used by a station set to request that the last party added to a conference be removed from the conference. If the call only involves two parties, the request is for a disconnect. The call or conference is that associated with the currently selected call appearance for the terminal. It is not necessary to specify this call appearance in the message.

The network will respond with a DROP ACKnowledge or DROP REJect message. If only two parties are involved, it will respond with a DISConnect.

Example

xds_att_l3_drop(16, 0x11);	this will drop b-channel 11 on board 16 from a connection or conference

xds_att_l3_drop_ack

int xds_att_l3_drop_ack(board_number, b_channel)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 DROP ACKnowledge message to a B-channel.

Message Sent
“DBxxDA”, where **xx** is the b-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a DROP ACKnowledge message to a selected b-channel on a selected board.

Example

xds_att_l3_drop_ack(16, 0x11);	this will send the drop ACK message to b-channel 11 on board 16

xds_att_l3_drop_rej

int xds_att_l3_drop_rej(board_number, b_channel)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 DROP REJect message to a B-channel.

Message Sent
"DBxxDR", where **xx** is the b-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a DROP REJect message to a selected b-channel on a selected board.

Example

xds_att_l3_drop_rej(16, 0x11);	this will send the drop REJect message to b-channel 11 on board 16

xds_att_l3_feature_ind

xds_att_l3_feature_ind(board_number, b_channel, button, module, type, feature, status)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char button;	the character 'U', 'C', 'F', or 'N', indicating the button type
char module;	a value between 0 and 7, indicating the module mode to use
char type;	the call type, valid characters are 'N', 'B', 'S', or 'M'
int feature;	a value between 0 and 255 specifying the feature indicator
char status;	a pointer to a character, between 0 and 7, indicating the indicator status (if any)

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 INFOrmation message with a Feature Indication element to control supplemental services. An optional indicator status may be included.

Message Sent
"DBxxFbmsffi", where **xx** is the B-channel number, **b** is the button type on the phone, **m** is the module mode used, **s** indicates the status type, **ff** is the feature number used, and **i** is the optional indicator status

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_TYPE	an invalid call type was selected
ILL_FEATURE	an invalid feature indicator value was passed
ILL_ARG	an invalid button type or module mode was used
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a INFORMATION message with a Feature Indication element to be sent. The Feature Indication element is used to control supplemental services, which are specific to a particular switch and network.

Valid values for the button type are:

U	unknown
C	call appearance button
F	feature button
N	non applicable

Valid values for the status type are:

N	feature number status
B	button number status
S	multiple button status
M	maintenance status

Valid values for the feature indicator status are:

0	active
1	deactivated
2	slow blink
3	medium blink
4	fast blink
5	solid, then off
6	fast blink, then off
7	pulse, then off

Example

xds_att_l3_feature_ind(16, 0x4, ‘N’, 1, ‘B’, 43, ‘0’);	this will send an INFORMATION message for B-channel 4, on board 16, with a button type of ‘N’, module mode of 1, a status type of ‘B’, feature number 43, with an optional “active” status message.
xds_att_l3_feature_ind(16, 0x4, ‘N’, 1, ‘B’, 43, 0);	this will send an INFORMATION message for B-channel 4, on board 16, with a button type of ‘N’, module mode of 1, a status type of ‘B’, feature number 43, with no optional status message.

xds_att_l3_feature_key

xds_att_l3_feature_key(board_number, b_channel, feature)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int feature;	a value between 0 and 63 specifying the feature number

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 INFOrmation message with a Feature number to control supplemental services.

Message Sent
"DBxxFff", where **xx** is the B-channel number and **ff** is the feature number used

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_FEATURE	an invalid feature key was used
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a INFORMATION message with a Feature number to be sent. The Feature number is used to control supplemental services which are specific to the particular switch and network.

Example

xds_att_l3_feature_key(16, 0x4, 63);	this will send an INFORMATION message for B-channel 4, on board 16, with a feature number of 63.

xds_att_l3_hold

xds_att_l3_hold(board_sw1, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 HOLD message to a B-channel.

Message Sent
"DBxxH" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD message to a selected b-channel on a selected board.

Example

xds_att_l3_hold_ack(1, 0x4);	this sends a HOLD message to B-channel 4 on board 1

xds_att_l3_hold_ack

xds_att_l3_hold_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 HOLD ACKnowledge message to a B-channel.

Message Sent
"DBxxHA" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD ACKnowledge message to a selected b-channel on a selected board.

Example

xds_att_l3_hold_ack(1, 0x4);	this sends a HOLD ACKnowledge message to B-channel 4 on board 1

xds_att_l3_hold_rej

xds_att_l3_hold_rej(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0x00 and 0xFF that indicates the reason for the hold rejection

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 HOLD REJect message to a B-channel.

Message Sent
"DBxxHRcc" where **xx** is the B-channel number, and **cc** is the cause for the rejection

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_CAUSE	an invalid hold rejection cause value was passed
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD REJect message to a selected b-channel on a selected board.

Example

xds_att_l3_hold_rej(1, 0x4, 0x7F);	this sends a HOLD REJect message to B-channel 4 on board 1, with a cause of 0x7F

xds_att_l3_mim_spid

xds_att_l3_mim_spid(board_number, b_channel, *spid);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *spid;	a pointer to a character array containing an ASCII string with the SPID (TE interfaces only) the SPID may be up to 14 digits

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This message is used by a terminal to send the Service Profile Identity (SPID) to identify the terminal and its capabilities to the network.

Message Sent
"DBxxMS" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
In the AT&T Custom Protocol a Management Information Message (MIM) is used to transfer the Service Profile Identity (SPID) rather than as an information element in an INFOrmation message. This may be done by a terminal either as part of the initialization procedure or in response to a MIM request from the network. Such a request would be indicated by a message of the form DBxxMS where **xx** is the b-channel when received by a port acting as a terminal. If the **spid** argument in the function call is a NULL string, the default SPID programmed on the board will be used. Otherwise, the argument string will be used. SPIDs are between 10 and 14 digits long depending on the switch. Typically, switches using the AT&T Custom protocol use a 10 digit SPID.

Example

xds_att_l3_mim_spid(1, 0x4, "608555200001");	this sends the message to B-channel 4 on board 1, with a SPID of "608555200001"

xds_att_l3_reconnect

xds_att_l3_reconnect(board_number, b_channel, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int reference;	the call reference number

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 reconnect message to a B-channel that is on hold.

Message Sent
"DBxxRrr" where **xx** is the B-channel number, and **rr** is the call reference number used

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to reconnect a channel that is on hold.

Example

xds_att_l3_reconnect(1, 0x4, 0x7F);	this sends a reconnect message to B-channel 4 on board 1, with a call reference number of 0x7F

xds_att_l3_reconnect_ack

xds_att_l3_reconnect_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 reconnect ACKnowledge message to a B-channel.

Message Sent
"DBxxRA" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to send a reconnect ACKnowledge message.

Example

xds_att_l3_reconnect_ack(1, 0x4);	this sends a reconnect ACKnowledge message to B-channel 4 on board 1

xds_att_l3_reconnect_rej

xds_att_l3_reconnect_rej(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	the cause number of the rejection

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 reconnect message to a B-channel that is on hold.

Message Sent
"DBxxRRrr" where **xx** is the B-channel number, and **rr** is the call reference number used

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_CAUSE	an invalid hold rejection cause value was passed
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to send a reconnect REJect message, with the cause.

Example

xds_att_l3_reconnect_rej(1, 0x4, 0x7F);	this sends a reconnect REJ message to B-channel 4 on board 1, with a call reference number of 0x7F

xds_att_l3_redirect

xds_att_l3_redirect(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used to send a Layer 3 redirect message to a B-channel that is on hold.

Message Sent
"DBxxK" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to reconnect a channel that is on hold.

Example

xds_att_l3_redirect(1, 0x4);	this sends a redirect message to B-channel 4 on board 1

xds_att_l3_switchhook

xds_att_l3_switchhook(board_number, b_channel, hook_state);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char hook_state;	the hook state value to be used, either ‘N’ or ‘F’

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is used by a terminal to inform the network of the switch-hook status of the terminal. It is valid only for ports acting as a terminal or user.

Message Sent
“DBxxSs” where **xx** is the B-channel number, and **s** is the hook state chosen

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid hook state value was called
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
In the AT&T Protocol, an INFOrmation message is used by a terminal to indicate whether the terminal is on-hook or off-hook. This is done as part of the initialization procedure and after a call has been completed and the terminal returns to the idle state.

The hook-state argument can be either ‘F’ for off-hook, or ‘N’ for on-hook.

Example

xds_att_l3_switchhook(1, 0x4, ‘N’);	this reports B-channel 4 on board 1 being ON-HOOK

xds_att_l3_text

xds_att_l3_text(board_number, b_channel, call_app, text_b, text_c);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int call_app;	a value between 0x0 and 0xFF indicating the text operation
char *text_b	a pointer to a NULL terminated string of up to 12 ASCII characters
char *text_c	a pointer to a NULL terminated string of up to 12 ASCII characters

Applicable boards
XDS BRI Boards

Purpose
This function is used to prepare text to be included in a Layer 3 message. Several modes are available that allow inclusion of text into a SETUP message, or provide for sending an INFOrmation message with text. Only NT (network termination) ports can send text.

Message Sent
"DBxxAaa" where **xx** is the B-channel, and **aa** is the call appearance

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid call appearance value or text string was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The display on the 8510T station set is broken up into three areas, a two character call appearance, a twelve character area for displaying a number, and a third area for displaying text. non-standard Codeset 6 information element that uses this format. The element includes coding to specify which of the three areas the text is to be displayed in. To accommodate this format, the XDS Layer 3 message to send text to a terminal from an NT port uses the form DBxxXaabbbbbbbbbbbbcccccccccccc where xx is the B-channel, aa is the call appearance, bbbbbbbbbbbb is a 12 character field for the number, and ccccccccccc is any additional text.

The number field must be padded with spaces to be exactly twelve characters long. As an example, the message:

Example

xds_att_l3_text(1, 0x4, 11, “6085551000”, “Caller One”);	this will display for B-channel 4 on board 1: the call appearance of 11, the number “6085551000” and “Caller One”.

xds_att_l3_transfer

xds_att_l3_transfer(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is to send a layer 3 transfer to a b-channel.

Message Sent
"DBxxT" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used during the call transfer stage of a call.

Example

xds_att_l3_tranfer(1, 0x4);	this send a transfer message to B-channel 4 on board 1

xds_att_l3_transfer_ack

xds_att_l3_transfer_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is to send a layer 3 transfer ACKnowledge message to a b-channel.

Message Sent
"DBxxTA" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to ACKnowledge a transfer being made.

Example

xds_att_l3_tranfer_ack(1, 0x4);	this sends a transfer ACKnowledge message to B-channel 4 on board 1

xds_att_l3_transfer_rej

xds_att_l3_transfer_rej(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value between 0 and 0xFF for the REJection cause

Applicable boards
AT&T Custom XDS BRI ISDN Boards

Purpose
This function is to send a layer 3 transfer REJect message to a b-channel.

Message Sent
"DBxxTRcc" where **xx** is the B-channel number and **cc** is the cause for the REJection.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_CAUSE	an invalid hold rejection cause value was passed
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to REJect a transfer from being made.

Example

xds_att_l3_tranfer_rej(1, 0x4, 0xFF);	this sends a transfer REJect message to B-channel 4 on board 1, with a cause of 255

XDS PCI-based MC3 Board Functions

This page was intentionally left blank.

xds_mc3_ring_mode

xds_mc3_ring_mode(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a value between 0 and 6 indicating the which mode the MC3 fiber should operate in

Applicable boards
Infinity Series H.100/H.110 MC3 boards

Purpose
This function is used to set the operating/ring mode for the MC3 fiber rings.

Message Sent
"SM0" and "SR0" for mode 0.
"SM1" and "SR0" for mode 1.
"SM2" and "SR0" for mode 2.
"SM2" and "SR1" for mode 3.
"SR2" for mode 4.
"SR1" for mode 5.
"SM1" and "SR2" for mode 6.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_ARG	an invalid mode was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

The **xds_mc3_ring_mode** is used to set the operating mode of the MC3 fiber rings. Three modes are supported. Mode 0 is the extended mode. Both rings are available providing a total of 4846 timeslots. In modes 1-6 the rings operate redundantly and 2423 timeslots are available.

The modes are:
0 - extended mode
1 - redundant, ring 0 primary
2 - redundant, ring 1 primary
3 - redundant, ring 0 primary, ring 0 failed, ring 1 becomes primary
4 - redundant. ring 0 primary, ring 1 failed
5 - redundant, ring 1 primary, ring 0 failed
6 - redundant, ring 1 primary, ring 1 failed, ring 0 becomes primary

Example

xds_mc3_ring_mode(16, 1);	this will set board 16 to operate in mode 1.

xds_ct_listen

xds_ct_listen(board_number, ct_xmt_timeslot, mc3_rcv_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int ct_xmt_timeslot;	a value between 0 and 4095 for the output H.100/H.110 timeslot
int mc3_rcv_timeslot;	a value between 0 and 4845 indicating which timeslot on the MC3 bus is serving as an input

Applicable boards
Infinity Series H.100/H.110 MC3 boards

Purpose
This function creates a one-way audio connection from an MC3 bus timeslot to a H.100/H.110 timeslot.

Message Sent
"XLIoooozzzz" where **oooo** is the H.100/H.110 timeslot and **zzzz** is the MC3 bus timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_SLOT	an invalid timeslot was chosen
WRONG_BOARD	an invalid board type was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function performs essentially half of an **xds_mc3_connec**t. Any of the 4846 MC3 bus timeslots may be connected to a port on the H.100/H.110 using this function. If the MC3 bus is configured as redundant rings, only 2423 timeslots are available. See **xds_mc3_connect** for details of the MC3 bus.

Example

xds_ct_listen(16, 2, 1535);	this creates an audio path from timeslot 1535 on the MC3 bus to timeslot 2 on the H.100/H.110 bus using board 16.

xds_ct_pattern

xds_ct_pattern(board_number, ct_xmt_timeslot, pattern);

unsigned char board_number;	a value indicating which board the command is for
int ct_xmt_timeslot;	a value between 0 and 4095 for the output H.100/H.110 timeslot
unsigned char pattern;	a value between 0x0 and 0xFF for the pattern

Applicable boards
Infinity Series H.100/H.110 MC3 boards

Purpose
This function may be used to output a fixed pattern on the H.100/H.110 bus. This may be used for diagnostic purposes or to place "silence" or some other pattern on a particular timeslot.

Message Sent
"XPIxxxxpp" where **xxxx** is the port number and **pp** is the pattern value.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_ARG	an invalid pattern was chosen
ILL_SLOT	an invalid timeslot was chosen
WRONG_BOARD	an invalid board type was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will cause the pattern value to be output to the H.100/H.110 timeslot, which is defined by the base timeslot set for the board at initialization and the port number defined by the ct_xmt_timeslot parameter. Placing a pattern on a timeslot may be used as a diagnostic tool to check bus integrity. It may also be used to place a "silence" pattern of 0xFF on the H.100/H.110 bus. A pattern may be disabled by using the **xds_ct_rls** function.

Example

xds_ct_pattern(16, 2, 0xFF);	this outputs the pattern 0xFF on the timeslot reserved for timeslot 2 on board 16.

xds_ct_rls

xds_ct_rls(board_number, ct_xmt_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int ct_xmt_timeslot;	a value between 0 and 4095 for the output H.100/H.110 timeslot

Applicable boards
Infinity Series H.100/H.110 MC3 boards

Purpose
This function will release a connection between the MC3 bus and the H.100/H.110 bus and disable the output on the H.100/H.110 bus timeslot reserved for the port to transmit on. This function will break down only half of a full duplex connection.

Message Sent
“XDIoooo” where **oooo** is the H.100/H.110 timeslot number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_SLOT	an invalid timeslot was chosen
WRONG_BOARD	an invalid board type was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The **xds_ct_rls** function releases a connection from the MC3 bus to the H.100/H.110 bus that has been established using the **xds_mc3_connect, xds_mc3_listen**, or **xds_ct_pattern** functions. To release a connection from the H.100/H.110 bus to the MC3 bus, the **xds_mc3_rls** function must be used. Therefore, to release a full duplex connection both the **xds_ct_rls** and **xds_mc3_rls** functions must be used.

Example

xds_ct_rls(16, 2);	this will disable the H.100 output from timeslot 2 on board 16.

xds_mc3_connect

**int xds_mc3_connect(board_number, ct_xmt_timeslot, ct_rcv_timeslot,
mc3_xmt_timeslot, mc3_rcv_timeslot);**

unsigned char board_number;	a value indicating which board the command is for
int ct_xmt_timeslot;	a value between 0 and 4095 for the H.100/H.110 output (transmit) timeslot
int ct_rcv_timeslot;	a value between 0 and 4095 for the H.100/H.110 input (receive) timeslot
int mc3_xmt_timeslot;	a value between 0 and 4845 for the MC3 output (transmit) timeslot
int mc3_rcv_timeslot;	a value between 0 and 4845 for the MC3 input (receive) timeslot

Applicable boards
Infinity Series H.100/H.110 MC3 boards

Purpose
This function creates a two-way connection between the H.100/H.110 bus and the MC3 bus. The MC3 bus can be used to connect multiple chassis.

Message Sent
"XCooooiiiiyyyyzzzz" where **oooo** is the H.100/H.110 bus output port, **iiii** is the H.100/H.110 bus input timeslot, **yyyy** is the MC3 bus output timeslot and **zzzz** is the MC3 bus input timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_SLOT	an invalid timeslot was chosen
WRONG_BOARD	an invalid board type was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The MC3 bus consists of two fiber optic rings, each of which has a capacity of 2423 timeslots. When operated in the extended mode, a total of 4846 timeslots are available. When operated in one of the redundant modes, only 2423 timeslots are available.

The **xds_mc3_connect** function is used to create a full-duplex connection between the local H.100/H.110 bus and the MC3 bus. The H.100/H.110 MC3 board at the other end of the connection must have a complementary connection established. The choice of MC3 bus timeslots is up to the application to select appropriate streams for transmitting on and to choose the specific timeslots to be used for each connection.

Example

xds_mc3_connect(16, 2, 103, 512, 1535);	this creates a connection between timeslot 1535 on the MC3 bus and timeslot 2 on the H.100/H.110 bus, and between timeslot 103 on the H.100/H.110 bus and timeslot 512 on the MC3 bus.

xds_mc3_listen

xds_mc3_listen(board_number, mc3_xmt_timeslot, ct_rcv_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int ct_xmt_timeslot;	a value between 0 and 4095 for the input H.100/H.110 timeslot
int mc3_rcv_timeslot;	a value between 0 and 4845 indicating which timeslot on the MC3 bus is serving as an output

Applicable boards
Infinity Series H.100/H.110 MC3 boards

Purpose
This function creates a one-way audio connection between a timeslot on the H.100/H.110 bus and a timeslot on the MC3 bus.

Message Sent
"XLOyyyyiiii" where **iiii** is the H.100/H.110 bus input timeslot and **yyyy** is the MC3 bus output timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_SLOT	an invalid timeslot was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The MC3 bus consists of two fiber optic rings, each of which has a capacity of 2423 timeslots. When operated in the extended mode, a total of 4846 timeslots are available. When operated in a redundant mode, only 2423 timeslots are available.

The **xds_mc3_listen** function is used to create a half-duplex connection between the local H.100/H.110 bus and the MC3 bus. The H.100/H.110 MC3 Board at the other end of the connection must have a complementary connection established.

Example

xds_mc3_transmit(16, 512, 102);	this creates an audio path from timeslot 102 on the H.100/H.110 bus on board 16 to timeslot 512 on the MC3 bus.

xds_mc3_pattern

xds_mc3_pattern(board_number, mc3_xmt_timeslot, pattern);

unsigned char board_number;	a value indicating which board the command is for
int mc3_rcv_timeslot;	a value between 0 and 4845 indicating which timeslot on the MC3 bus is serving as the output
unsigned char pattern;	a value between 0x0 and 0xFF to be output to the MC3 bus

Applicable boards
Infinity Series H.100/H.110 MC3 boards

Purpose
This function may be used to output a fixed pattern on the MC3 bus. This may be used for diagnostic purposes or to place a "silence" pattern on the MC3 bus.

Message Sent
"XPOyyyypp" where **yyyy** is the MC3 bus timeslot and **pp** is the pattern value.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid pattern was chosen
ILL_SLOT	an invalid timeslot was chosen
WRONG_BOARD	an invalid board type was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The MC3 bus consists of two fiber optic rings, each of which has a capacity of 2423 timeslots. When operated in the extended mode, a total of 4846 timeslots are available. When operated in a redundant mode, only 2423 timeslots are available.

The **xds_mc3_pattern** function may be used to operate a fixed value on a timeslot on the MC3 bus. A pattern, once established may be disabled by using the **xds_mc3_rls** function.

Example

xds_mc3_pattern(16, 512, 0x45);	this outputs the pattern 0x45 on timeslot 512 of the MC3 bus from board 16.

xds_mc3_loopback_mode

xds_mc3_loopback_mode(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a value between 0 and 0xF indicating the loopback mode for the MC3 fiber

Applicable boards

Infinity Series H.100/H.110 MC3 boards

Purpose

This function is used to set the loopback mode for the MC3 fiber rings. This can be used for diagnostic or testing purposes.

Message Sent

"SLx" where **x** is the mode in hexadecimal.

Returns

This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid mode was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

The **xds_mc3_loopback_mode** is used to set the loopback mode of the MC3 fiber rings. Available modes are from 0 to 15. This can be used for testing purposes. Each of four bits in the mode value control one of the loopback functions.

These bits are:
0 TLBB - Terminal Loopback B
1 FLBB - Facilities Loopback B
2 TLBA - Terminal Loopback A
3 FLBA - Facilities Loopback A

In normal operation, all bits should be 0. For more details on the loopback operation see board's hardware reference manual.

Example

xds_mc3_loopback_mode(16, 1);	this will enable TLBB on board 16.

xds_mc3_rls

xds_mc3_rls(board_number, mc3_xmt_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int mc3_rcv_timeslot;	a value between 0 and 4845 indicating which timeslot on the MC3 bus is serving as the output

Applicable boards
Infinity Series H.100/H.110 MC3 boards

Purpose
This function will release a connection between the H.100/H.110 bus and the MC3 bus and disable the output on the MC3 bus timeslot. This function will break down only half of a full duplex connection.

Message Sent
"XDOyyyy" where **yyyy** is the MC3 bus transmit timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	invalid timeslot
WRONG_BOARD	invalid board type
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The **xds_mc3_rls** function releases a connection from the H.100/H.110 bus to the MC3 bus that has been established using the **xds_mc3_connect** or **xds_mc3_liste**n, or **xds_mc3_pattern** functions. To release a connection from the MC3 bus to the H.100/H.110 bus, the **xds_ct_rls** function must be used. Therefore, to release a full duplex connection both the **xds_ct_rls** and **xds_mc3_rls** functions must be used.

Example

xds_mc3_rls(16, 512);	this will disable the output from board 16 on timeslot 512 of the MC3 bus.

xds_mc3_2k_tone

xds_mc3_2k_tone_det(board_number, ct_rcv_timeslot, mode);

unsigned char board_number;	a value indicating which board the command is for
int ct_rcv_timeslot;	a value between 0 and 4095 for the H.100/H.110 timeslot
char mode;	a value indicating whether to enable or disable generation

Applicable boards
Infinity Series H.100/H.110 MC3 boards

Purpose
This function will either enable or disable SS7 2 kHz tone generation on a H.100/H.110 timeslot.

Message Sent
"CToooоE" where **oooo** is the H.100/H.110 timeslot that the tone will be outputted on.
"CTooooD" where **oooo** is the H.100/H.110 timeslot to disable the tone on.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_SLOT	invalid timeslot
WRONG_BOARD	invalid board type
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The generator should not be enabled if any of the 2 kHz. detectors are enabled as this may result in a possible CCA assignment conflict. If a conflict occurs, an error message of the form **EG01** will be reported. The generator may be disabled with a command of the form **CTooooD** where **oooo** is the H.100/H.110 bus stream and timeslot. In practice, it is best to enable the generator at start up time and leave it connected to the H.100/H.110 bus.

Example

xds_mc3_2k_tone(16, 512, 'E');	this will enable 2 kHz tone generation on timeslot 512 of board 16
xds_mc3_2k_tone(16, 512, 'D');	this will disable 2 kHz tone generation on timeslot 512 of board 16

xds_mc3_2k_tone_det

xds_mc3_2k_tone_det(board_number, ct_rcv_timeslot, mode);

unsigned char board_number;	a value indicating which board the command is for
int ct_rcv_timeslot;	a value between 0 and 4095 for the H.100/H.110 timeslot
char mode;	a value indicating whether to enable or disable detection

Applicable boards
Infinity Series H.100/H.110 MC3 boards

Purpose
This function will either enable or disable SS7 2 kHz tone detection on a given timeslot.

Message Sent
"CLiiii" where **iiii** is the H.100/H.110 timeslot to enable detection on.
"CLiiiiD" where **iiii** is the H.100/H.110 timeslot to disable detection on.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_SLOT	invalid timeslot
WRONG_BOARD	invalid board type
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Signaling System 7 (SS7) uses a 2 kHz. tone for performing continuity checks to verify the operation of speech circuits. The H.100/H.110 MC3/Conferencing Board is capable of generating and detecting this tone. A single generator is provided to play a 2 kHz. tone to an H.100/H.110 bus timeslot. From there, it may be routed to multiple MC3 bus timeslots. Up to 128 detectors are available for the detection of the 2 kHz tones. The DSP resources for 2 kHz. detection are shared with those used for DTMF detection and clamping of conferenced inputs. Conferencing **must** be enabled for the 2 kHz. detection and generation to be available. Note, that at most, 128 detectors, generators and conference parties can be assigned at a time.

Example

xds_mc3_2k_tone_det(16, 512, 'E');	this will enable detection on timeslot 512, on board 16
xds_mc3_2k_tone_det(16, 512, 'D');	this will disable detection on timeslot 512, on board 16

This page was intentionally left blank.

H.100/110 Line Board Common Functions

This page was intentionally left blank.

xds_line_ct_listen

int xds_line_ct_listen(board_number, port, stream, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating the port to use for the listen
int stream;	a value indicating the H.100 stream to use for the listen
int timeslot;	a value indicating the H.100 timeslot to use for the listen

Applicable Boards
XDS Infinity H.100 & H.110 line-interface boards

Purpose
This function will create a one-way audio interface between the CT bus timeslot and the port on the indicated board.

Message Sent
"CAxxsstt" where **xx** is the port, **ss** is the CT bus stream, and **tt** is the CT bus timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_SLOT	an invalid CT bus stream and/or timeslot was called
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will create a one-way audio interface between the CT bus timeslot and the port on the indicated board.

Example

xds_line_ct_listen(16, 0, 2, 1);	directs timeslot 1 of stream 2 to port 0 on board 16

xds_line_ct_connect

int xds_line_ct_connect(board_number, port, output_steam, output_timeslot
input_steam, input_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating the port to connect
int output_stream;	the output stream to connect
int output_timeslot;	the output timeslot to connect
int input_stream;	the input stream to connect
int input_timeslot;	the input timeslot to connect

Applicable Boards
XDS Infinity H.100 & H.110 line-interface boards

Purpose
This function is used to connect an input stream and timeslot with an output stream and timeslot to a port interface.

Message Sent
"CCxxssttaabb" where **xx** is the port, **ss** is the output stream, **tt** is the out put timeslot, **aa** is the input stream, and **bb** is the input timeslot

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
ILL_SLOT	an invalid stream or timeslot was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The valid range of streams is from 0 to 31, while the valid range of timeslots is 0 to 127.

Example

xds_line_ct_connect(16, 11, 0, 3, 10, 50);	connects input stream 0 timeslot 3 and output stream 10 timeslot 50 to port 11, on board 16

xds_line_ct_play_call_progress

int xds_line_ct_play_call_progress(board_number, tone, stream, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int tone;	a value between 0 and 5 indicating which call progress tone to play
int stream;	a value indicating which CT bus stream to use
int timeslot;	a value indicating which CT bus timeslot to use

Applicable Boards
XDS Infinity H.100 & H.110 line-interface boards

Purpose
This function can be used to play any of the six precise call progress tones to a port. These tones are dial tone, reorder, busy, and audible ringback. In addition, silence and a calibration tone of 1004 Hz. can be played at a level corresponding to 1 milliwatt.

Message Sent
“CS2xsstt” where **x** is the tone number, **ss** is the CT bus stream, and **tt** is the CT bus timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an illegal stream and/or timeslot value was passed
ILL_MODE	an invalid call progress tone was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comment
There is no limit on the number of ports listening to any of the call progress tones.

The six tones produced by this function are selected with the tone argument. The tones produced are:

0 - dial tone
1 - reorder tone
2 - busy tone
3 - audible ringback
4 - digital milliwatt 1004 Hz steady tone.
5 - silence

Example

xds_line_ct_play_call_progress(16, 5, 3, 10);	this will play "silence" to stream 3, timeslot 5 of board 16

xds_line_ct_generate_customtone

xds_line_ct_generate_customtone(board_number, port, frequency1, frequency2, level1 level2, on_dur, off_dur, reps);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
int frequency1;	a value between 0 and 0x0CFF (0 to 3327 Hz) for the frequency of the first tone (in Hz), a value of 0 disables the tone
int frequency2;	a value between 0 and 0x0CFF (0 to 3327 Hz) for the frequency of the second tone (in Hz), a value of 0 disables the tone
int level1;	a value between 0 and 0x3F giving the level in -dBm of the first tone
int level2;	a value between 0 and 0x3F giving the level in -dBm of the second tone
int on_dur;	a value between 0 and 0xFF giving the duration in 50 msec. increments of the on portion of the tone
int off_dur;	a value between 0 and 0xFF giving the duration in 50 msec. increments of the off portion of the tone
int reps;	a value between 1 and 255 giving the number of times the tone repeats

Applicable Boards
XDS Infinity H.100 & H.110 line-interface boards

Purpose
This function can be used to generate a custom tone for user alerts or other purposes. The tone can be made up of either a single frequency or dual frequencies. The duration of both the on and off portion of the tone can be controlled, as well as the number of repetitions.

Message Sent
"CVxxffffllffffllnnffrr" where **xx** is the port, **ffff** is the frequency of the first and second tones, **ll** is the level of the first and second tones, **nn** is the on duration, **ff** is the off duration, and **rr** is the number of repetitions.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_ARG	an out of range value was used for one of the arguments
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function uses the DTMF generators to generate the custom tone.

The custom tones produced with this function can consist of one or two frequencies. The frequencies should be within the telephone frequency range of 0 to 3327 Hz. If only a single frequency is desired, the other frequency should have a value of 0. The level of the frequencies can be specified in a range of 0 dBm to -62 dBm, with a larger level value producing a quieter tone. If the frequency is not to be used, the level value should be set to 63.

The duration of the ON portion of the tone is given by on_dur in 50 msec. increments in a range of 50 msec to 12700 milliseconds. A value of 0xFF for the duration variable will play a continuous tone.

Example
xds_line_ct_generate_customtone(16, 2, 1000, 1350, 10, 15, 8, 5, 3);

This will cause a tone consisting of 1000 Hz. at -10 dBm and 1350 Hz at -15 dBm. to be repeated 3 times on port 2 with an ON duration of 400 msec and an OFF duration of 250 msec.

xds_line_ct_set_transmit

xds_line_ct_set_transmit(board_number, port, stream, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
int stream;	a value indicating the stream number to use
int timeslot;	a value indicating the timeslot number to use

Applicable Boards
XDS Infinity H.100 & H.110 line-interface boards

Purpose
This function will create a one-way audio interface between the port and a CT bus timeslot. The port will be seized by this function.

Message Sent
"CXxxsstt" where **xx** is the port number, **ss** is the stream, and **tt** is the timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_SLOT	an invalid CT bus timeslot and or stream was selected for use
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The timeslot the port will transmit on is determined by the port number and the base timeslot of the board set at initialization in accordance with the procedure spelled out in the 4.1 SC specification. Any audio path created by a previous command will be disabled.
The board will respond with a state change message of sub-type 14.

Example

xds_line_ct_set_transmit(16, 2, 0, 1);	this will cause port 2, stream 0 and timeslot 1, of board 16 to transmit to the CT bus

xds_line_ct_reset_lineport

xds_line_ct_reset_lineport(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the port that indicates the port interface

Applicable boards
XDS Infinity H.100 & H.110 line-interface boards

Purpose
This function is used to reset a port.

Message Sent
"RPxx" where **xx** is the port number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function can be used to force a reset of a port.

Example

xds_line_ct_reset_lineport(16, 4);	this will reset port 4 on board 1

xds_line_ct_set_custom_dialtone

xds_line_ct_set_custom_dialtone(board_number, port, frequency1, frequency2, level1 level2, on_duration, off_duration);

unsigned char board_number;	a value indicating which board the command is for
int tone;	a value between 0 and 0x1A indicating which tone to use
int frequency1;	a value between 0 and 0x0CFF (0 to 3327 Hz) for the frequency of the first tone (in Hz), a value of 0 disables the tone
int frequency2;	a value between 0 and 0x0CFF (0 to 3327 Hz) for the frequency of the second tone (in Hz), a value of 0 disables the tone
int level2;	a value between 0 and 0x3F giving the level in -dBm of the second tone
int level1;	a value between 0 and 0x3F giving the level in -dBm of the first tone
int on_duration;	a value between 0 and 0xFF giving the duration in 50 msec. increments of the on portion of the tone
int off_duration;	a value between 0 and 0xFF giving the duration in 50 msec. increments of the off portion of the tone

Applicable Boards
XDS Infinity H.100 & H.110 line-interface boards

Purpose
This function can be used to generate a custom tone for user alerts or other purposes. The tone can be made up of either a single frequency or dual frequencies. The duration of both the on and off portion of the tone can be controlled, as well as the number of repetitions.

Message Sent
"SDttffffllffffllnnff" where **tt** is the tone, **ffff** is the frequency of the first and second tones, **ll** is the level of the first and second tones, **nn** is the on duration, and **ff** is the off duration.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_ARG	an out of range value was used for one of the arguments
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function uses the DTMF generators to generate the custom tone.

The custom tones produced with this function can consist of one or two frequencies. The frequencies should be within the telephone frequency range of 0 to 3327 Hz. If only a single frequency is desired, the other frequency should have a value of 0. The level of the frequencies can be specified in a range of 0 dBm to -62 dBm, with a larger level value producing a quieter tone. If the frequency is not to be used, the level value should be set to 63.

The tone can be repeated, with the number of repetitions given by the reps value. This must be within the range 1 to 255. A value of 0 is not valid. The duration of the ON portion of the tone is given by on_dur in 50 msec. increments in a range of 50 msec to 12700 milliseconds. A value of 0xFF for the duration variable will play a continuous tone.

Example
xds_line_ct_set_custom_dialtone(16, 2, 1000, 1350, 10, 15, 8, 5, 3);

This will cause a tone consisting of 1000 Hz. at -10 dBm and 1350 Hz at -15 dBm. to be repeated 3 times on port 2 with an ON duration of 400 msec and an OFF duration of 250 msec.

xds_line_ct_set_custom_inftone

xds_line_ct_set_custom_inftone(board_number, port, frequency1, frequency2, level1 level2, on_duration1, on_duration2, off_duration1, off_duration2);

unsigned char board_number;	a value indicating which board the command is for
int tone;	a value between 0 and 0x1A indicating which tone to use
int frequency1;	a value between 0 and 0x0CFF (0 to 3327 Hz) for the frequency of the first tone (in Hz), a value of 0 disables the tone
int level1;	a value between 0 and 0x3F giving the level in -dBm of the first tone
int frequency2;	a value between 0 and 0x0CFF (0 to 3327 Hz) for the frequency of the second tone (in Hz), a value of 0 disables the tone
int level2;	a value between 0 and 0x3F giving the level in -dBm of the second tone
int on_duration1;	a value between 0 and 0xFF giving the duration in 50 msec. increments of the on portion of the 1st tone
int on_duration2;	a value between 0 and 0xFF giving the duration in 50 msec. increments of the on portion of the 2nd tone
int off_duration1;	a value between 0 and 0xFF giving the duration in 50 msec. increments of the off portion of the 1st tone
int off_duration2;	a value between 0 and 0xFF giving the duration in 50 msec. increments of the off portion of the 2nd tone

Applicable Boards
XDS Infinity H.100 & H.110 line-interface boards

Purpose
This function can be used to generate a custom tone for user alerts or other purposes. The tone can be made up of either a single frequency or dual frequencies. The duration of both the on and off portion of the tone can be controlled, as well as the number of repetitions.

Message Sent
"SIttffffllffffllnnffnnff" where **tt** is the tone, **ffff** is the frequency of the first and second tones, **ll** is the level of the first and second tones, **nn** is the on duration for the first and second tones, and **ff** is the off duration for the first and second tones.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_ARG	an out of range value was used for one of the arguments
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function uses the DTMF generators to generate the custom tone.

The custom tones produced with this function can consist of one or two frequencies. The frequencies should be within the telephone frequency range from 0 to 3327 Hz. If only a single frequency is desired, the other frequency should have a value of 0. The level of the frequencies can be specified in a range of 0 dBm to -62 dBm, with a larger level value producing a quieter tone. If the frequency is not to be used, the level value should be set to 63.

The tone can be repeated, with the number of repetitions given by the reps value. This must be within the range 1 to 255. A value of 0 is not valid. The duration of the ON portion of the tone is given by on_dur in 50 msec. increments in a range of 50 msec to 12700 milliseconds. A value of 0xFF for the duration variable will play a continuous tone.

Example
xds_line_ct_set_custom_inftone(16, 2, 1000, 10, 1350, 15, 8, 5, 3);

This will cause a tone consisting of 1000 Hz. at -10 dBm and 1350 Hz at -15 dBm. to be repeated 3 times on port 2 with an ON duration of 400 msec and an OFF duration of 250 msec.

xds_line_ct_set_gain

int xds_line_ct_set_gain(board_number, port, xmt_polarity, xmt_gain, rcv_polarity, rcv_gain);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char xmt_polarity;	'+' positive gain, '-' negative gain
short xmt_gain;	a value between 0 and 10 indicating the amount of gain in 1 dBm steps
char rcv_polarity;	'+' positive gain, '-' negative gain
short rcv_gain;	a value between 0 and 10 indicating the amount of gain in 1 dBm steps

Applicable Boards
XDS Infinity H.100 & H.110 line-interface boards

Purpose
This function is used to control the gain associated with each port on the board.

Message Sent
"SGxxpttprr" where **xx** is the port is the port, the first **p** is the polarity of the transmit gain, **tt** is the transmit gain level, the second **p** is the polarity of the receive gain, and **rr** is the receive gain level.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_MODE	an invalid polarity was selected for use
ILL_ARG	an out of range value was used for one of the gain arguments
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
There may be situations because of loop length where it is desirable to add gain or attenuation to a port. This can be done with this command. The polarity can be either a "+" or a "-". The gain is specified in 1 dBm steps with a range from -10 to + 10 dBm.

Example

xds_line_ct_set_gain(16, 2, '+', 6, '-', 8)	this sets port 2 on board 16 for 6 dB of transmit gain and -8 dB receive gain

xds_line_ct_set_hookflash

int xds_line_ct_set_hookflash(board_number, port, min_time, max_time);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating which port is being controlled
unsigned char min_time;	the minimum amount of time to detect a hookflash
unsigned char max_time;	the maximum amount of time to detect a hookflash

Applicable Boards
XDS Infinity Loop, E&M, and Station boards

Purpose
This function is used to set the hookflash detect times for a specified port.

Message Sent
"SHxxaabb" where **xx** is the port is the port, aa = minimum time, bb = maximum time.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-Infinity Station board was used
ILL_PORT	an invalid port was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The timing of the flash or pause signal may vary with switch type and country. In North America the "flash" signal is normally between 350 and 1000 msec. In Europe the "pause" signal is shorter, between 50 and 100 msec. To allow for these variations, the hook-flash timing can be changed on a port by port basis. On-hook signals of less than the minimum time will be ignored, those that are greater than the maximum will be treated as a disconnect. Note that this only affects ports set to type "**phone**".

The hook flash timing parameters may be saved in the EEROM.

Example

xds_line_ct_set_hookflash(16, 2, 28, 4B);	this would set the minimum time for port 2 to 400 msec. and the maximum time to 750 msec.2 of board 16

xds_em_ct_lead_control

int xds_em_ct_lead_control(board_number, port, mode);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char mode;	a value indicating the relay mode, ‘O’ - open, ‘C’ - close

Applicable Boards
XDS Infinity H.100 E&M interface boards

Purpose
This function is used to control a pair of relays associated with each port on this special interface board.

Message Sent
“XCxx” closes relay for port **xx**
“XOxx” opens relay for port **xx**

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_MODE	an invalid relay mode was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function controls the relays on a special interface board developed for connecting to radio systems. There is also an audio interface that is identical to that on an E&M board. No ‘E’ or ‘M’ lead control signals are supported.

Example

xds_em_ct_lead_control(16, 2, ‘C’)	this closes relay for port 2 on board 16
xds_em_ct_lead_control(16, 2, ‘O’)	this opens relay for port 2 on board 16

xds_em_ct_guard_tone

int xds_em_ct_guard_tone(board_number, port, channel_number);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char channel_number;	a value indicating the relay mode, 'O' - open, 'C' - close

Applicable Boards
XDS Infinity H.100 E&M boards

Purpose
This function is used to play a guard tone on a port for a given channel number or remove the guard tone.

Message Sent
"CGxxt" plays low-level guard tone on port **xx** for channel number **t**
"CGxxX" removes low-level guard tone for port **xx**

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_MODE	an invalid channel number was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command will only work if the port is in the Connect or MVIP connect state It will cause a 2175 HZ tone followed by a tone that selects the channel number. The valid tones are: '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F'. They correspond to channel numbers 1 – 16. To remove the tone is 'X'.

Example

xds_em_ct_guard_tone(16, 2, '0')	this plays a guard tone followed by a tone for channel number 1 on port 2 for board 16.
xds_em_ct_guard_tone(16, 2, '1')	this plays a guard tone followed by a tone for channel number 2 on port 2 for board 16.
xds_em_ct_guard_tone (16, 2, 'X')	this removed the guard tone on port 2 for board 16.

xds_line_ct_set_signaling_protocol

int xds_line_ct_set_signaling_protocol(board_number, port, incoming_protocol, digit_format, address_digit_count, optional_outgoing_protocol);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char incoming_protocol;	a value indicating the protocol for incoming calls
char digit_format;	a value indicating whether the digits are sent by DTMF or pulses
char address_digit_count;	a value indicating number of digits
char optional_outgoing_protocol;	an optional value indicating the protocol for outgoing calls

Applicable Boards
XDS Infinity H.100 E&M and Station interface boards

Purpose
This function is used to set the signaling protocol for a port.

Message Sent
E&M board:
"SPxxidn(o)" where **xx** is the port number, **i** indicates the protocol for incoming calls, **d** indicates whether the digits are sent by DTMF or pulses, **n** is the number of digits, and **o** indicates the protocol (if any) for outgoing calls.

Station board:
"SPxxidn" where **xx** is the port number, **i** indicates the protocol for incoming calls, **d** indicates whether the digits are sent by DTMF or pulses, and **n** is the number of digits.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
WRONG_BOARD	a board other than an E&M or Station board has been selected
ILL_ARG	an invalid incoming or outgoing call protocol was chosen
ILL_TYPE	an invalid digit format was chosen
ILL_MODE	an invalid number of digits was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
For the Station board, the **optional_outgoing_protocol** will not be used and the function should use a NULL argument for this option.

The options for **incoming_protocol** are

E&M board:	Station board:
‘D’ - delay dial	‘I’ - immediate start
‘I’ - immediate start	‘W’ – wink start
‘W’ – wink start	

The options for **digit_format** are

E&M board:	Station board:
‘P’ – pulse dial	‘P’ – pulse dial
‘T’ – tone dial	‘T’ – tone dial
	‘M’ – MF

The options for **address_digit_count** are from 1 – 15 for the number of digits

The options for **optional_outgoing_protocol** (*optional) are

E&M board:	Station board:
‘D’ - delay dial	NULL – not used
‘I’ - immediate start	
‘W’ –wink start	
NULL – not used	

Example
xds_line_ct_set_signaling_protocol(16, 0, ‘W’, ‘P’, 4, ‘D’)

this sets port 0 for 4 DTMF digits, wink start, pulse dialing, with delay dial and an outgoing protocol of delay dial on board 16

xds_line_ct_set_signaling_protocol(16, 0, ‘W’, ‘P’, NULL)

this sets port 0 for 4 DTMF digits, wink start on board 16

xds_loop_ct_port_options

int xds_loop_ct_port_options(board_number, port, option_a, option_b, option_c);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char option_a;	a value indicating the first option mode
char option_b;	a value indicating the second option mode
char option_c;	a value indicating the third option mode
char option_d;	a value indicating the fourth option mode

Applicable Boards
XDS Infinity H.100 & H.110 Loop-start Interface boards

Purpose
Sets options for a given Loop-start board port.

Message Sent
"SOxxabc" where **xx** is the port, **a** is the first option, **b** is the second option, **c** is the third option, and **d** is the fourth option.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_MODE	an invalid first, second, third, or fourth option mode was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

Sets the options o for port xx. Each option may be set to 'Y' for yes, 'N' for no, or '*' for no change. The default is no. If option **a** is set to 'Y' (yes), an "SF" message will be sent if loop current is detected upon seizure of the loop. If option **b** is set to 'Y' (yes), changes in battery polarity will be reported with "Sr" and "Sn" messages. If option **c** is set to 'Y' (yes), the line will not be seized, but an audio path will be created when a *connect* or other command is issued. If option **d** is set to 'Y' (yes), the port will not be disconnected on a linebreak. If option **d** is set to 'N' (no), then the port will be disconnected on a linebreak.

Examples

xds_loop_ct_port_options(17, 0, ‘Y’, ‘N’, ‘Y’, ‘N’)

this will send an “SF” message if loop current is detected upon seizure of the loop and the line will not be seized, but an audio path will be created when a *connect* or other command is issued, and the port will be disconnected on a linebreak on port 0 on board 17

xds_loop_ct_port_options(17, 2, ‘N’, ‘Y’, ‘N’, ‘*’)

this will report “Sr” and “Sn” messages when there are changes in battery polarity
on port 2 on board 17

This page was intentionally left blank.

H.100/110 Station Board Functions

This page was intentionally left blank.

xds_station_ct_call_waiting

int xds_station_ct_call_waiting(board_number, port, tone);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating which port is being controlled
int tone;	a value between 0 and 4 indicating which call progress tone to play

Applicable Boards
XDS Infinity Station boards

Purpose
This function can be used for the call waiting feature on the boards.

Message Sent
"CWxxt" where **xx** is the port number and **t** is the tone.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-Infinity Station board was used
ILL_ARG	an invalid tone was selected
ILL_PORT	an invalid port number was called
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

A tone can be issued to a port that is already in the connect state to indicate that another call is waiting. This tone consists of a 440 Hz. signal in one of several patterns. The command to send this tone takes the form **"CWxxp"** where xx is the port number and p is the pattern. The pattern codes are:

1 - a single tone for 300 msec.
2 - two tone bursts of 100 msec. each with 100 msec. Between
3 - three tone bursts of 300 msec. each with 100 msec. Between
4 - 100 msec. tone, 300 msec., and 100 msec. Tones

To respond to this tone, the user sends a hook-flash signal. The application can then put the first call on hold and connect the user to the second caller.

Example

xds_station_ct_call_waiting(16, 2, 3);	this will play tone 3 to port 2 of board 16

xds_station_ct_return_polarity

int xds_station_ct_return_polarity (board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating which port is being controlled

Applicable Boards
XDS Infinity Station boards

Purpose
This function is used to return the polarity on a given port back to normal.

Message Sent
“PNxx” where **xx** is the port.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-Infinity Station board was used
ILL_PORT	an invalid port number was called
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The polarity may only be reversed for ports that are not in the idle or ringing states. The polarity will automatically return to normal when a port is disconnected or goes on-hook.

Example

xds_station_ct_return_polarity(16, 2);	this will return the polarity on port 2 for board 16 back to normal

xds_station_ct_reverse_polarity

int xds_station_ct_reverse_polarity(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating which port is being controlled

Applicable Boards
XDS Infinity Station boards

Purpose
This function is used to reverse the polarity on a given port.

Message Sent
"PRxx" where **xx** is the port.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-Infinity Station board was used
ILL_PORT	an invalid port number was called
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The polarity may only be reversed for ports that are not in the idle or ringing states. The polarity will automatically return to normal when a port is disconnected or goes on-hook.

Example

xds_station_ct_reverse_polarity(16, 2);	this will reverse the polarity on port 2 for board 16

xds_station_ct_set_calling_name

int xds_station_ct_set_calling_name(board_number, port, calling_name);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating which port is being controlled
char *calling_name;	a pointer to the calling name to be displayed

Applicable Boards
XDS Infinity Station boards

Purpose
This function is used to display a calling party's name.

Message Sent
"DNxx<name>" where **xx** is the port is the port and **name** is the name to be displayed.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-Infinity Station board was used
ILL_ARG	an invalid (NULL) calling name was selected
ILL_PORT	an invalid port was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The calling party name may be added to the information with a command of the form **"DNxxname"** where **xx** is the port and name is the calling party name. The board will accept up to 28 characters in the name field, but current devices are limited to either 15 or 21 characters. In the case where the name is not available or private, the name may either be omitted or replaced with a "?O" if not available or "?P" if private. The "DN" command must be sent after the "DD" command.

Example
xds_station_ct_set_calling_name(16, 2, "John Smith");

this sets port 2 on board 16, with "John Smith" calling

xds_station_ct_set_date_calling_number

int xds_station_ct_set_date_calling_number(board_number, port, date, calling_num);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating which port is being controlled
char *date;	a pointer to the date to be displayed
char *calling_num;	a pointer to the calling number to be displayed

Applicable Boards
XDS Infinity Station boards

Purpose
This function is used to display a date along with a calling party's number.

Message Sent
"DDxxmmddhhmm/#" where **xx** is the port is the port, **mmdd** is the month and date, **hhmm** is the hour and minute, and **#** is the calling party number to be displayed.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-Infinity Station board was used
ILL_ARG	an invalid (NULL) calling number was selected
ILL_PORT	an invalid port was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
To send caller identity information, the time, date, and number must be set up before the ringing or call waiting command is issued. This is done with a message of the form "**DDxxmmddhhmm/#"** where **xx** is the port number, **mmdd** is the month and date, **hhmm** is the hour and minute, and **#** is the calling party number. The date and time are in decimal using a 24 hour format. The calling party number should be between 4 and 10 digits. Longer numbers may not display properly. In some cases, it may desired not to display the number either because it is not available or restricted. In this case, the number may be replaced with an "O" for out of area if not available, or with a "P" for private.

Example
xds_station_ct_set_date_calling_number(16, 2, "04030201", "6085551234");

this will display "April 3, 2:01 AM, 608-555-1234" on port 2 of board 16

xds_station_ct_set_port_profile

int xds_station_ct_set_port_profile(board_number, profile);

unsigned char board_number;	a value indicating which board the command is for
char *profile;	a pointer to a string of characters which sets the profile for all of the ports on the board

Applicable Boards
XDS Infinity Station boards

Purpose
This function is used to set the hookflash detect times for a specified port.

Message Sent
“SPpp….p” where **p** is the port profile to be used for the respective port

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-Infinity Station board was used
ILL_ARG	an invalid profile / profile string was selected
ILL_PORT	an invalid port was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Each port can be set to the required characteristics by selecting the appropriate parameter profile. Ten different profiles numbered 0-9 are provided. Six of these are fixed, while four can be customized for specific needs. The profiles are selected using the **"SP"** command. This command consists of "SP" followed by the profile number of each of the 24 ports. The profiles are:

0 - default does not use CRAM coefficients
1 - North American, 20 Hz. internal ringing
2 - North American, 30 Hz. internal ringing
3 - North American, 20 Hz. external ringing
4 - European, 25 Hz. internal ringing
5 - European, 25 Hz. external ringing
6 - User profile 1
7 - User profile 2
8 - User profile 3
9 - User profile 4

Example
xds_station_ct_set_port_profile(16, "111111111111222222222222";

this would set the port profile for the first 12 ports to the North American 20 Hz. Internal ring, and the last 12 ports to the North American 30 Hz. internal ring

xds_station_ct_set_profile_parms

int xds_station_ct_set_profile_parms(board_number, profile_num, reg, parms);

unsigned char board_number;	a value indicating which board the command is for
char profile_num;	a character of the selected profile
int reg;	the selected register
char *parms;	a pointer to a string of characters which sets the parameters for a specified profile

Applicable Boards
XDS Infinity Station boards

Purpose
This function is used to set the custom port profiles.

Message Sent
"SPVprr(vv…v)" where **p** is the port profile to be configured, **rr** is the register number to be used, and the **v**'s are the desired values for the port profiles.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-Infinity Station board was used
ILL_ARG	an invalid register or an invalid/NULL parameter string was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function takes a bit of knowledge in programming custom profiles, therefore you will need to refer to the *H.100 Station Board Manual* (Amtelco P/N 257M003).

Example
xds_station_ct_set_profile_parms(16, 6, 70, "4A0C2324C4922403");

xds_station_ct_set_visual_message_waiting

int xds_station_ct_set_visual_message_waiting(board_number, port, enable);	
unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating which port is being controlled
char enable;	a value of ‘F’ or ‘N’ indicating the enable mode to be used

Applicable Boards
XDS Infinity Station boards

Purpose
This function is used to turn the *Visual Message Waiting Indicator* on or off.

Message Sent
“DVxxF” where **xx** is the port
or
“DVxxN” where **xx** is the port

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-Infinity Station board was used
ILL_PORT	an invalid port was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Visual Message Waiting Indicator is controlled using a command of the form **“DVxxa”** where **xx** is the port and a is the action, either “F” to turn the indicator on or “N” to turn the indicator off. This command can only be issued when the port is in the on-hook idle condition.

Example

xds_station_ct_set_visual_message_waiting(16, 1, ‘F’);

this will turn Visual Message Waiting Indicator off on port 1 of board 16

xds_station_ct_set_voltage

int xds_station_ct_set_voltage(board_number, v1, v2);

unsigned char board_number;	a value indicating which board the command is for
char v1;	a value between 0 and 6
char v2;	a value between 0 and 7

Applicable Boards
XDS Infinity Station boards

Purpose
This function is used to set the voltage levels of both fixed and user profiles on a board.

Message Sent
"SVab" where **xx** is the port

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-Infinity Station board was used
ILL_ARG	invalid voltage values were chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
These are the available voltage levels for each board:
0 - automatic detection
1 - 28 V
2 - 48 V
3 - 72 V
4 - 96 V
5 - 120 V
6 - 144 V
7 - user value (programmable profiles only)

Example

xds_station_ct_set_voltage(16, 1, 2);	this will set the voltage level for fixed profiles to 28 Volts and user profiles to 48Volts on board 16.

This page was intentionally left blank.

ISA / PCI / cPCI Common Line-interface Board Functions

This page was intentionally left blank.

xds_line_cptones

int xds_line_cptones(board_number, port, tone);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
int tone;	a value between 0 and 5 indicating which call progress tone to play

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
This function can be used to play any of the six precise call progress tones to a port. These tones are dial tone, reorder, busy, and audible ringback. In addition, silence and a calibration tone of 1004 Hz. can be played at a level corresponding to 1 milliwatt.

Message Sent
"CPxxt" where **xx** is the port number and **t** is the tone.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_ARG	an invalid call progress tone was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comment
Using **xds_line_cptones** will seize a port unless that port is an E&M or DID port that has is in the off-hook waiting or listen state. In those states, the seizure condition will remain unchanged. To force a seizure, **xds_line_hold** may be called before the **xds_line_cptones.** Any existing audio path to or from this port will be disabled by this function. There is no limit on the number of ports listening to any of the call progress tones.

The six tones produced by this function are selected with the tone argument. The tones produced are:

0 - dial tone
1 - reorder tone
2 - busy tone
3 - audible ringback
4 - digital milliwatt 1004 Hz steady tone.
5 - silence

The first four tones are as defined in RS-464.

Example

xds_line_cptones(1, 2, 3);	this will play "busy tone" to port 2 of board 1

xds_line_detect_energy

int xds_line_detect_energy(board_number, port, duration);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
int duration;	a value between 0 & 224 indicating the number of .1 of second energy must persist before being reported, if 0, detection is disabled

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
This function will set a port to detect audio energy. This can be used for detecting call progress tones or voice to determine whether a call has been answered.

Message Sent
"CExxdd" where **xx** is the port and **dd** is the duration in the range 20 - FF.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_ARG	an invalid value for the duration was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The energy detect function can be used to detect whether a call has been answered, or to detect call progress signals either before or after dialing. Any connection that exists at the time detection is enabled will remain. If the port is not transmitting when the command is issued, it will be placed in the transmit state. If detection is disabled using the **xds_line_detect_energy** function with the duration set to 0, any connection will not be affected. This means that if the port is in the transmit state, it will remain in that state. Detection will also be disabled by any command that changes the state of the port, for example a connect or a hold function.

The minimum duration that the energy must persist before being reported is set using the duration value, which is in tenths of a second. When a signal has lasted for this amount of time it will be reported. Cessation of the signal will also be reported. It is not necessary to re-enable detection after a signal has been detected. Thus, cadences of regularly repeating signals can be determined. Also, the duration can be used to discriminate between signals of different duration. Thus, by setting the duration to 4, the call progress signal busy which consists of .5 seconds of tone followed by .5 seconds of silence will be detected and reported, while reorder, which consists of .25 sec. of tone followed by .25 sec. of silence will not be reported.

Energy detection will be reported in a message of type 1, sub type 15. If the argument is 1, energy has been detected, if the argument is 0, a detected signal has stopped.

Examples

xds_line_detect_energy(1, 2, 4)	this will detect energy of .4 seconds or greater duration on port 2 of board 1.
xds_line_detect_energy(1, 2, 0)	will disable energy detection on port 2 of board 1

xds_line_disconnect

int xds_line_disconnect(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
This function will release a port. The line will be released and the transmit timeslot will be disabled.

Message Sent
"CDxx" where **xx** is the port number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The port will be returned to the idle state by this function. However, E&M ports and station ports set to type **phone** will not be returned to the idle state until after they have gone on-hook in response to the port release. A state change message of sub-type 7 will be returned by the board in response to this function.

Example

xds_line_disconnect(1, 2);	this will disconnect port 2 of board 1

xds_line_hold

int xds_line_hold(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
This function will put the port in the hold state. The port will be seized but audio to and from the port will be disabled.

Message Sent
"CHxx" where **xx** is the port number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Using **xds_line_hold** will seize a port unless that port is an E&M or Station port that has been issued an **xds_line_disconnect** and has not yet gone on-hook to return to the idle state. The board will respond to an **xds_line_hold** with a state change message of sub_type 6.

There are three valid ways for a port to reach the hold state. The first is in response to a call to the **xds_line_hold** function. The second is in response to a call to the **xds_line_seize** function if the port is in the idle state. The third is at the termination of sending a tone string.

Example

xds_line_hold(1, 2);	this will place port 2 on board 1 in the hold state

xds_line_hookflash

xds_line_hookflash(board_number, port, mode, time);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char mode;	a value specifying what mode of time interval to use for the hookflash (10 millisecond steps or 100 millisecond steps)
int time;	a value between indicating the duration in .1 sec. or .01 sec, depending on the mode, of the hookflash

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
This function will send a hook-flash out of the B-channel. The duration of the hook-flash is determined in either .1 second or .01 second increments by the **time** argument, depending on the mode selected. Hook-flash signals may be used to get the attention of a PBX or other phone system for performing some action such as transferring a call.

Message Sent
"CFxxd" where **xx** is the B-channel number and **d** is the duration in .1 second increments.
"CFxxdd" where **xx** is the B-channel number and **dd** is the duration in .01 second increments.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_MODE	an invalid time interval mode was selected
ILL_ARG	an invalid value for the hookflash time was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The range of hook-flashes that may be sent is .1 to 1.5 seconds or .01 to 2.5 seconds. If current is not flowing in the line circuit, this command will cause a port to hangup. No state change response message is sent for this function.

Example

xds_t1e1_hookflash(16, 2, 1, 7);	will cause a hook-flash of .7 seconds on port 2 of board 16
xds_t1e1_hookflash(16, 2, 2, 7);	will cause a hook-flash of .07 seconds on port 2 of board 16

xds_line_disable_output

xds_line_disable_output(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
This function will disable audio output to the SCbus in those states where an idle pattern may be output. These states are hold, listen, off-hook waiting, or idle-pattern. In the listen state, the port will change to the hold state. In the idle pattern state, the port will change to the idle state. In the other states, the state of the port will not change. Calling this function for a port in any other state will have no effect.

Message Sent
“CIxx” where **xx** is the port number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
When a port changes from the connect or transmit state to the hold state, or when an on-hook condition is detected in either of those states, an idle pattern of silence will be output to the SCbus. It may be desirable in these cases to disable the output to the SCbus. This is done with the **xds_line_disable_output** function. The states will change as indicated above, with the appropriate state change message being sent by the board. In the 4.1 SC model, transmit timeslots are reserved for each port, so there is no need to free a timeslot with an idle pattern on it. Therefore, there is no benefit to using this function under this model.

Example

xds_line_disable_output(1, 2);	will disable the idle pattern for port 2.

xds_line_state

xds_line_state(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
This function will return the current line state of a port on an analog line board or of a B-channel on the Basic Rate ISDN Board. This information may be used for diagnostic purposes.

Message Sent
None.

Returns
This function will return with the port state code. If there is no board with the proper board number or the port is not a valid port on that board, a (-1) will be returned.

Comments
This function obtains information by reading a table in the dual-ported ram. This table is synchronized to an internal table. However, there may be cases where there are differences between the two.

The defined states for the analog line boards are:

0 - idle
1 - connected
2 - on hold
3 - ringing being detected
4 - transmitting, audio to the SCbus
5 - ringing being generated
6 - listening, audio from the SCbus or an internal source
7 - off-hook waiting to proceed
8 - conferenced (using resources local to the board only)
9 - off-hook waiting to go idle
10 - idle pattern - generating an idle pattern to the SCbus, behavior the same as idle

For the SCSA U-Interface Basic Rate ISDN Board, the line states reflect the state of the B-channels and not the call state. Note, that the state information is not available for the S/T-Interface board. The states for the SCSA BRI Boards are:

0 - idle, no audio path
1 - connect
2 - hold
3 - transmit, one way audio to the SCbus
4 - listen, audio from the SCbus or an internal source
5 - detecting, a DTMF or energy detector connected to the B-channel

Example

xds_line_state(1, 2);	this requests the state of port 2 on board 1, the function will return the state.

xds_line_detect_dtmf

int xds_line_detect_dtmf(board_number, port, mode);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char mode;	a character indicating listen mode is to be used

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
This function will set a port to detect DTMF digits. Dial-tone will be generated until the first digit is detected unless the "Quiet" option is chosen, in which case, no dial-tone will be played. The 'M' or "Monitor" option can be used to detect digits while the port is in the connect state. As each DTMF tone is detected, a state change message of sub-type 13 will be sent by the board with the argument indicating the tone.

Message Sent
"CLxx" where **xx** is the port number or "CLxxo" where **o** indicates an option.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The DTMF detector is capable of detecting all 16 of the standard DTMF tones, 0-9, *, #, A-D. Each board is equipped with eight detectors, one for each line. Once a detector is assigned, it remains assigned until a command is issued for that port or an on-hook is detected.

The DTMF detect function will disable any audio paths to or from the port at the time it is called. If connected, the audio path will be broken and must be reestablished after the digits are detected. The port will be seized by this function.

If the monitor mode is used by selecting the '**M**' option, existing connections or one-way audio paths will be maintained. The state of the port will also remain unchanged. Because connections are maintained, no dial-tone is played to the port. Digits detected will be reported in the usual manner. Detection may be stopped by using the **xds_line_detect_dtmf** function with the mode set to **'F'**. Detection will also be stopped if the port changes state due to a command or change in hook status. The monitor mode is only supported in firmware versions 2.0 and higher.

if ' ', dial tone is played until the first DTMF tone is detected
if '**Q**', no dial tone is played,
if '**M**', DTMF is monitored without affecting existing connections
if '**F**', DTMF detection is turned off

Example

xds_line_detect_dtmf(1, 2, ' ');	this will set port 2 of board 1 to listen for DTMF tones, dial tone will be played until the first digit is detected

The board will respond with a message of type 1, subtype 9**,** port number 2. As each DTMF digit is detected, the board will return a message of type 1, subtype 13, port number 2. The argument will be the digits detected encoded as an ASCII character.

xds_line_detect_dtmf(1, 2, 'Q');	this will do the same as the first example, except dial tone will not be played.

xds_listen_dtmf(1, 2, 'M');	this will turn on detection but maintain existing connections
xds_listen_dtmf(1, 2, 'F');	this will turn off detection while maintaining existing connections

xds_line_seize

xds_line_seize(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
The purpose of this function is to seize an idle port and place it in the hold state. If a port is not idle, it will not be seized and a "busy" state change will be returned. This function can be used to minimize "glare" or collisions between incoming and outgoing calls.

Message Sent
"CBxx" where **xx** is the port number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

(-1)	the port is busy
ILL_PORT	an interface port was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function may be used for reducing the possibility of "glare" by calling the function before making an outgoing call on a port. If the port is busy because an incoming call has been detected, the function will return a (-1) and the board will return a "SBxx" message. An additional state change message, such as ring detect or off-hook, indicating the incoming call on that port should also be received.

Example

xds_line_seize(1, 2);	this will seize port 2 on board 1

xds_line_send_dial_string

xds_line_send_dial_string(board_number, port, tones);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char *tones;	a pointer to an ASCII string of tones to be played

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
This function will cause a string of DTMF tones to be sent out a port. The tone string takes the form of a NULL terminated ASCII string. The port will be seized by this function. In addition to the standard DTMF tones, the string may consist of pauses and single frequency tones.

Message Sent
"CTxx<string>" where **xx** is the port.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Each board has DTMF generators. If an attempt is made to generate more tone strings than possible by a board, an error will be returned from this function.

The allowed tones may vary from board to board, so please refer to your hardware manual for a list of these.

Example

xds_line_send_dial_string(1, 2, "123P456");	this will cause the tone string "123P456" to be sent out port 2 of board 1.

xds_xmt_timeslot

xds_xmt_timeslot(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone

Applicable Boards
XDS line-interface and BRI-interface boards

Purpose
This function will retrieve the base timeslot for a given port.

Message Sent
None

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will retrieve the base timeslot for a given port. This function is helpful in application timeslot assignment routines.

Example

xds_xmt_timeslot(1, 2);	this will return the transmit timeslot for port 2

This page was intentionally left blank.

SCSA ISA
Line Board Functions

This page was intentionally left blank.

xds_listen

int xds_listen(board_number, port, stream, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should do the auditing
int stream;	an SC bus stream number indicating which timeslot to be audited
int timeslot;	an SC bus timeslot number indicating which timeslot to be audited

Applicable Boards
XDS SCSA line-interface and BRI-interface boards

Purpose
This function will create a one way audio interface between the SCbus timeslot and the port on the indicated board. The port will be seized by this function unless the port is an E&M port in the off-hook state, in which case the audio path will be created but the M lead will not be asserted.

Message Sent
"CAxxstt" where **xx** is the port number, **s** is the stream number, and **tt** is the timeslot

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_SLOT	an invalid SC bus stream and/or timeslot was called
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The board will respond with a state change message of type 1, sub-type 9.

Example

xds_listen(1, 7, 1, 0)	this listens to timeslot 0, on stream 1 with port 7 on board 1

xds_connect

**int xds_connect(board_number, port, input_stream, input_timeslot,
output_stream, output_timeslot);**

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should be connected
int input_stream;	an SC bus input stream number
int input_timeslot;	an SC bus input timeslot number
int output_stream;	an SC bus output stream number
int output_timeslot;	an SC bus output timeslot number

Applicable Boards
XDS SCSA line-interface and BRI-interface boards

Purpose
The purpose of this command is to establish a two-way audio connection between the port and another port or device. The port will transmit on a timeslot determined by the base timeslot of the board and the port. This may be determined by using **xds_xmt_timeslot.** The port will receive on the timeslot value in the function call.

Message Sent
"CCxxsttabb" where **xx** is the port number, **s** is the input stream, **tt** is the input timeslot, **a** is the output stream, **bb** is the output timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_SLOT	an invalid SC bus stream and/or timeslot was called
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
To connect two ports, a connect function must be called for both ports.

Using **xds_connect** will seize a port unless that port is an E&M or Station port that has been issued an **xds_disconnect** and has not yet gone on-hook to return to the idle state. An **xds_connect** may be issued for a port already in the connect state without going through any intermediate state.

Example

```
xds_connect(0, 0, 1, 2, 0, 0);  make connection on board 0, port 0
xds_connect(1, 2, 0, 0, 1, 2);  make connection on board 1, port 2
```

xds_monitor

int xds_monitor(board_number, port, input_stream, input_timeslot,
output_stream , output_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
int input_stream;	an SC bus input stream number
int input_timeslot;	an SC bus input timeslot number
int output_stream;	an SC bus output stream number
int output_timeslot;	an SC bus output timeslot number

Applicable Boards
XDS SCSA Monitor board

Purpose
This function is used to set a port on the SCSA Monitor Board to monitor two SCbus timeslots.

Messages Sent
"CMxxsttabb" where **xx** is the port number, and **stt** & **abb** are the two sets of streams and timeslots being monitored.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_SLOT	an invalid SC bus stream and/or timeslot was called
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Monitor Board uses the DSP to combine two timeslots into a single output so that a port on this board may be used to simultaneously monitor both sides of a conversation. The first port on this board is a normal station port to which a standard phone instrument may be connected. The second port consists of a line level input and an amplified output suitable for driving a speaker. Each port can be set to independently monitor pairs of timeslots. The first port can also act as a standard station port for receiving or placing calls if needed.

Example
xds_monitor(0, 1, 123, 234)   this would set port 1 on board 0 to monitor timeslots 123 and 234

xds_mwi

xds_mwi(board_number, port, mwi_state);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char mwi_state;	a character to indicate which state the MWI indicator should be in an 'N' indicates on and an 'F' indicates off.

Applicable Boards
XDS SCSA Station board

Purpose
This function will enable or disable the Message Waiting Indicator feature of the Station Board for a specific port. This feature allows a light or LED to be illuminated on a phone equipped with such a device. This indicator can be used to display whether messages are waiting for that station.

Message Sent
"WNxx" to turn on the indicator for port **xx**.
"WFxx" to turn off the indicator for port **xx**.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_MODE	an MWI mode was called
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This feature of the board requires that a ring generator and -48 volt power source be connected to the board. It also requires properly equipped telephone sets to function, as the indicator is part of the set. The indicator will go off when the station is activated or goes off-hook, but will return to the proper state when the station returns to the idle condition.

Example

xds_mwi(1, 2, 'N');	this will turn the indicator feature on for port 2 of board 1

xds_relay_control

int xds_relay_control(board_number, port, relay, mode);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char relay;	a value of 0x0 or 0x1 indicating which relay is affected
char mode;	a value indicating the relay mode, 0 = open, 1 = closed

Applicable Boards
XDS SCSA Radio Interface board.

Purpose
This function is used to control a pair of relays associated with each port on this special interface board.

Message Sent
“XCxxr” closes relay r for port **xx**
“XOxxr” opens relay r for port **xx**

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_ARG	an invalid relay number was selected
ILL_MODE	an invalid relay mode was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function controls the relays on a special interface board developed for connecting to radio systems. Each port has two relays. There is also an audio interface that is identical to that on an E&M board. No E or M lead control signals are supported.

Example

xds_relay_control(1, 2, 0, 1)	this closes relay 0 for port 2 on board 1
xds_relay_control(1, 2, 0, 0)	this opens relay 0 for port 2 on board 1

xds_ring

xds_ring(board_number, port, cadence);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
int cadence;	a value between 0 and 3 indicating which ring cadence to use

Applicable Boards
XDS SCSA Station board

Purpose
This function will cause ringing to appear on a station port set to type "Phone". One of three different ring cadences may be selected by this function. Ringing will cease when the port goes off-hook or the **xds_disconnect** function is called for the port.

Message Sent
"CRxxc" where **xx** is the port number and **c** is the cadence code.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_MODE	an invalid cadence ring mode was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will only work with ports on a Station board that have been set to type “Phone” using the **xds_set_type** function. The port must be on-hook. It is also necessary that a ring generator and a -48V power source be connected to the board.

The allowed ring cadences are:
0 - default, 2 seconds on, 4 seconds off
1 - standard, 2 seconds on, 4 seconds off
2 - PBX ringing, 1 second on, 3 seconds off
3 - split ringing, 1 second on, 1 second off, 1 second on, 3 seconds off
No response message is sent by the board.

Example
xds_ring(1, 2, 3);   this will cause split ringing to be generated on port 2 of board 1.

xds_send_pulses

xds_send_pulses(board_number, port, digits);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char *digits;	a pointer to an ASCII string of digits to be pulsed

Applicable Boards
XDS SCSA Loop Start and E&M line boards

Purpose
This function will cause a string of digits to be pulsed out a port. The digit string takes the form of a NULL terminated ASCII string. The port will be seized by this function. In addition to the digits 0 through 9, the string may contain short or long pauses.

Message Sent
"COxx<string>" where **xx** is the port.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_ARG	an empty digit string was passed to the function
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The allowed digits consist of the digits 0-9 (0 causes 10 pulses). In addition, a long or short pause may be part of the string. Pulses are generated at a rate of 10 per second with a 700 msec. interdigit interval. The allowed ASCII characters and the tones they produce are:

0 – 9, the digits 0 - 9
X - short pause, 200 msec.
P - long pause, 2 seconds
If the port is idle the board will respond with a state change message of sub-type 6 when the string begins, otherwise, no state change will occur. A state change message of sub-type 4 will be sent when the string is completed.

Example

xds_send_pulses(1, 2, “123P456”);	this will cause the digit string “123P456” to be sent out port 2 of board 1.

The board will respond with a message of type 1, subtype 6, port number 2.
The digits 1, 2, and 3 will be sent followed by two second pause and then the digits 4, 5, and 6.
The board will send a message of type 1, subtype 4, port number 2 to indicate the string is completed.

xds_send_tones

xds_send_tones(board_number, port, frequency_1, level_1, frequency_2 ,level_2, on_dur, off_dur, reps);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
int frequency_1;	a value between 300 and 3000 for the frequency of the first tone in Hz., a value of 0 disables the tone
int level_1;	a value between 0 and 63 giving the level in -dBm of the first tone
int frequency_2;	a value between 300 and 3000 for the frequency of the second tone in Hz., a value of 0 disables the tone
int level_2;	a value between 0 and 63 giving the level in -dBm of the second tone
int on_dur;	a value between 0 and 80 giving the duration in 50 msec. increments of the on portion of the tone
int off_dur;	a value between 0 and 80 giving the duration in 50 msec. increments of the off portion of the tone
int reps;	a value between 1 and 255 giving the number of times the tone repeats

Applicable Boards
XDS SCSA line-interface boards

Purpose
This function can be used to generate a custom tone for user alerts or other purposes. The tone can be made up of either a single frequency or dual frequencies. The duration of both the on and off portion of the tone can be controlled, as well as the number of repetitions.

Message Sent
"CVxxffffllffffllnnoorr" where **xx** is the port, **ffff** is the frequency of the first and second tones, **ll** is the level of the first and second tones, **nn** is the on duration, **oo** is the off duration, and **rr** is the number of repetitions.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_ARG	an out of range value was used for one of the arguments
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function uses the DTMF generators to generate the custom tone. Each board has four DTMF generators.. If an attempt is made to generate more than four tone strings at a time an error message of type 5, subtype 2 will be returned by the board.

The custom tones produced with this function can consist of one or two frequencies. The frequencies should be within the telephone frequency range of 300 to 3000 Hz. If only a single frequency is desired, the other frequency should have a value of 0. The level of the frequencies can be specified in a range of 0 dBm to -62 dBm, with a larger level value producing a quieter tone. If the frequency is not to be used, the level value should be set to 63.

The tone can be repeated, with the number of repetitions given by the reps value. This must be within the range 1 to 255. A value of 0 is not valid. The duration of the ON portion of the tone is given by on_dur in 50 msec. increments in a range of 50 msec to 4 seconds. The range for the OFF or silent portion of the tone is the same. A duration of 0 for either parameter is not valid. The board will respond with a state change message of sub-type 9 when the tone begins. A state change message of sub-type 4 will be sent when the tone is completed.

Example
xds_send_tones(1, 2, 1000, 10, 1350, 15, 8, 5, 3);

This will cause a tone consisting of 1000 Hz. at -10 dBm and 1350 Hz at -15 dBm. to be repeated 3 times on port 2 with an ON duration of 400 msec and an OFF duration of 250 msec.

xds_set_agc

int xds_set_agc(board_number, port, xmt_mode, xmit_gain, rcv_mode, rcv_gain);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char xmt_mode;	AGC mode in the transmit direction, 'A' - AGC on, '+' positive gain, '-' negative gain
int xmit_gain;	a value between 0 and 40 indicating the amount of gain in .5 dB steps
char rcv_mode;	AGC mode in the receive direction, 'A' - AGC on, '+' positive gain, '-' negative gain
int rcv_gain;	a value between 0 and 40 indicating the amount of gain in .5 dB steps

Applicable Boards
XDS SCSA Radio Interface and Monitor boards

Purpose
This function is used to control the Automatic Gain Control (AGC) function of the DSP. associated with each port on the board.

Message Sent
"SAxxmttmrr" where **xx** is the port is the port, **m** is the AGC mode, **tt** is the transmit gain level and **rr** is the receive gain level.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_MODE	an invalid AGC mode was selected for use
ILL_ARG	an out of range value was used for one of the gain arguments
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Radio Interface and Monitor Boards have special DSP firmware that includes an AGC function. This function can be set independently on each port in each direction. It can operate in either an AGC mode, or can be used to introduce a fixed amount of gain or loss up to 20 dB.

Example

xds_set_agc(1, 2, '+', 6, '-', 8)	this sets port 2 on board 1 for 3 dB of transmit gain and -4 dB receive gain

xds_set_backplane_level

xds_set_backplane_level(board_number, level);

unsigned char board_number;	a value indicating which board the command is for
int level;	the backplane level in .5 dB steps, the range is +25 to -25

Applicable Boards
XDS SCSA line-interface boards

Purpose
This function is used to set the backplane level for a board. The transmit and receive levels are both changed from the 0 dB reference by the number of .5 dB steps given in the command.

Message Sent
"SRsll" sets the backplane level, **s** is the sign and **ll** is the level in .5 dB steps.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an out of range value was used for the backplane level
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Defining the backplane level of a board is part of the SCSA model. The intention is to allow the levels of a board to be shifted relative to the 0 dB level to accommodate the differing characteristics of different national standards for inputs such as T1 or E1 spans. This command allows the level to be set in .5 dB steps in a range from +12.5 to -12.5 dB. If the backplane level selected is beyond the capabilities of the board a response message of the type 5, subtype 6 for transmit level and subtype 7 for receive level will be sent by the board. In this case, the board level will be set to the maximum or minimum level possible.

Example

xds_set_backplane_level(1, -3);	this will set the backplane levels for board 1 to -1.5 dB.

xds_set_port_level

xds_set_port_level(board_number, port, xmt_level, rcv_level);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
int xmt_level;	the transmit level in .5 dB steps, the range is from +50 to -50
int rcv_level;	the receive level in .5 dB steps, the range is from +50 to -50

Applicable Boards
XDS SCSA line-interface boards

Purpose
This function is used to set the transmit and receive levels for a particular port. These level changes are referenced to the backplane level. The level parameters are given in the number of .5 dB steps and cover a range of +25dB to -25dB. This range may exceed the capabilities of the board.

Message Sent
"SLxxsttsrr" sets the levels for port **xx**; s is the sign, **tt** is the transmit level and **rr** is the receive level.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_ARG	an out of range value was used for one of the levels
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is intended to allow the setting of gain or loss for a particular port in both the transmit or receive directions. The intention is to allow the levels of a port to be shifted relative to the backplane level to accommodate the differing characteristics of lines connected to the port. This command allows the level to be set in .5 dB steps in a range from +25 to -25 dB. If the level selected is beyond the capabilities of the board a response message of the type 5, subtype 8 for transmit level and subtype 9 for receive level will be sent by the board. In this case, the level will be set to the maximum or minimum level possible.

Example

xds_set_port_level(1, 2 - 3, +2);	this will set the transmit level for board 1, port 2 to -1.5 dB and the receive level to +1 dB.

xds_set_protocol

xds_set_protocol(board_number, port, iwink, pls_tn, no_digits, owink, ani);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
char iwink;	a character indicating whether wink or immediate start 'W' or 'I'
char pls_tn;	a character indicating whether pulse or DTMF, 'P' or 'T'
int no_digits;	the number of address digits expected
char owink;	a character indicating whether outgoing calls are wink or immediate start
char ani;	a character indicating which ANI protocol to use

Applicable Boards
XDS SCSA DID and E&M line boards

Purpose
This function is used to set the protocol for a DID or E&M port. By setting the protocol, the board can be set to automatically handle all of the details of accepting incoming address digits or outgoing wink confirmation signals. It can also be used to select an ANI protocol on an E&M port.

Message Sent
"SPxxabcde" sets the protocol for port **xx** to parameters **a**-**e**.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_MODE	an invalid protocol was selected for use
ILL_ARG	the number of digits passed is greater than 15
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Direct-Inward-Dialing or DID, is a method of sending address digits from a switch to customer premise equipment. The digits may be sent as either a train of pulses or DTMF digits. The sender may require a wink acknowledgement before sending the digits. A port on a DID or E&M board may be set to automatically respond with any acknowledgement and collect the digits without requiring intervention from an application. DNIS, Dialed Number Identification System, or ANI, Automatic Number Identification are related systems that can be accommodated by the boards.

Five parameters are required to define the protocol on XDS boards. These are:
iwink - an 'I' or 'W' indicating whether the behavior is immediate or wink start on incoming calls
pls_tn - a 'P' or 'T' indicating whether address digits are sent as pulses or DTMF tones
no_digits - the number of address or DNIS digits expected
owink - an 'I' or 'W' indicating whether outgoing calls are immediate or wink start; only E&M ports can make outgoing calls
ANI - the ANI protocol, (only available on E&M). Currently, two ANI formats are supported, 'A' and 'B'. We believe that 'A' is appropriate for MCI and 'B' is appropriate for Sprint. As of this time, AT&T does not appear to offer ANI in a DTMF format.
When address digits are detected by a DID or E&M port, the board will respond with a state change message of sub-type 3.

Example
xds_set_protocol(1, 2, 'W', 'P', 3, 'I', ' ');   this will set up port 2 of board 1 with a DID protocol of incoming wink start, and 3 pulse address digits. No ANI protocol is indicated.

An incoming call would be indicated with a message of type 1, subtype 3, port 2, with the msg_str being a string with the DID address digits. This string would take the form:

"123" where 123 are the received address digits.

xds_set_protocol(1, 2, 'W', 'T', 4, 'I', 'A'); this will set up port 2 of board 1 with a protocol of incoming wink start, DTMF digits, 4 DNIS digits, immediate start on outgoing calls and type 'A' ANI format.

An incoming call would be indicated with a message of type 1, subtype 3, port 2, with the msg_str being a string with the address and ANI digits. This string would take the form:

"1234*9876543210" where 1234 is the DNIS or called party number and 9876543210 is the ANI or calling party number.

xds_set_volume

int xds_set_volume(board_number, volume);

unsigned char board_number;	a value indicating which board the command is for
int volume;	a value between 0 and 127 selecting the amplifier volume level.

Applicable Boards
XDS SCSA Monitor board.

Purpose
This function is used to control the volume on the monitor port amplifier.

Message Sent
“MVll” where **ll** is the level in hexidecimal (00-7F)

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an out of range value was used for the volume
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The output of port 1 of the Monitor Board is directed to an amplifier that can be used to drive a speaker. This command controls the level of the speaker. Because the output is non-linear and depends on the speaker connected, the volume is set in arbitrary steps in the range 0-127 with 127 being maximum output. It will be necessary for the application to determine which number of steps is needed to produce the desired output

Example

xds_set_volume(1, 44)	this sets the volume at level 44

xds_transmit

xds_transmit(board_number, port, stream, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port should receive the tone
int stream;	a value indicating the stream number to use
int timeslot;	a value indicating the timeslot number to use

Applicable Boards
XDS SCSA line-interface and BRI-interface boards

Purpose
This function will create a one-way audio interface between the port and an SCbus timeslot. The port will be seized by this function.

Message Sent
"CXxxstt" where **xx** is the port number, **s** is the stream, and **tt** is the timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_SLOT	an invalid SC bus timeslot and or stream was selected for use
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The timeslot the port will transmit on is determined by the port number and the base timeslot of the board set at initialization in accordance with the procedure spelled out in the 4.1 SC specification. Any audio path created by a previous command will be disabled.
The board will respond with a state change message of sub-type 14.

Example

xds_transmit(1, 2, 0, 1);	this will cause port 2, stream 0 and timeslot 1, of board 16 to transmit to the SC bus

This page was intentionally left blank.

Common Enhanced Conference Board Functions (SCSA Library Version)

This page was intentionally left blank.

xds_ct_cnf_timeslot

xds_ct_cnf_timeslot (board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value indicating the conference port outputting to the CT bus

Applicable boards
H.100/H.110 boards with conferencing resources

Purpose
This function returns the transmit timeslot for a specified conference port.

Message Sent
None

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-PCI based board was used for this function
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The number of available conference ports on each board varies, so please consult the board's reference manual for this number.

Example

xds_ct_cnf_timeslot (16, 4);	This would return the transmit timeslot of conference port 4 on board 16.

xds_ct_conference

xds_ct_conference(board_number, handle, ct_xmt_stream, ct_xmt_timeslot, ct_rcv_stream, ct_rcv_timeslot, atten, threshold);

unsigned char board_number;	a value indicating which board the command is for
char handle;	a value indicating which conference handle to assign
int ct_xmt_stream;	a value indicating which CT bus stream is serving as an output
int ct_xmt_timeslot;	a value indicating which CT bus timeslot is serving as an output
int ct_rcv_stream;	a value indicating which CT bus stream is serving as an input
int ct_rcv_timeslot;	a value indicating which CT bus timeslot is serving as an input
char atten;	a value between 0-7 that controls the input and output attenuation of the conference on the specified timeslots
char threshold;	a value between 0-3 specifying a noise threshold below which the input will not be added to a conference

Applicable boards
H.100/H.110 boards with conferencing resources and the SCSA Enhanced conference board

Purpose
This function adds a CT bus connection to the conference with the specified handle. The connection consists of a pair of CT bus timeslots, one for an input and one for an output. The attenuation and noise threshold parameters used by the conferencing hardware are specified.

Message Sent
"CAhhooooiiiiat" where **hh** is the conference handle, **oooo** is the CT bus output stream and timeslot, **iiii** is the CT bus input stream and timeslot, **a** is the attenuation parameter and **t** is the noise threshold parameter.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an illegal input and/or output stream/timeslot was used for the board
ILL_HANDLE	an invalid conference handle was chosen for the conference
ILL_ATTEN	an invalid attenuation value was chosen for the conference
ILL_THRESHOLD	an invalid threshold value was chosen for the conference
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Each conference board is equipped with a different number of conference bridges, varying on board type and density. Each group of conferences share a set of outputs called "Conference Control Addresses" or CCA's. To best determine the limit, please consult the hardware reference manual. The attenuation parameter can be used to attenuate the conference input, output, or both, in 3 dB steps. The following levels of attenuation can be achieved using these attenuation parameter values:

value	input attenuation	output attenuation
0	0 dB	0dB
1	0 dB	3 dB
2	3 dB	0 dB
3	3 dB	3 dB
4	6 dB	0 dB
5	6 dB	3 dB
6	9 dB	0 dB
7	9 dB	3 dB

As more parties are added to a conference, it may be necessary to increase the amount of attenuation to produced acceptable audio performance. The noise threshold parameter sets a threshold below which an input signal is not added to a conference. A value of 0 sets it for no threshold, while 3 sets the threshold to the highest value. The higher levels of noise threshold will introduce increasing levels of distortion and so should be used with caution. Typical values for small conferences are 0 for both the attenuation and noise threshold parameters. An **xds_ct_conference** call may be issued for a connection already conferenced to change the attenuation and noise threshold parameters for that connection. The **xds_conference_parameter** function can be used to change the parameters for all parties to a conference. It is the responsibility of the application to select a conference handle and to keep track of which handles are in use and which parties are assigned to each handle.

Valid values for the transmit streams and receive streams are 0 – 31. Valid values for the transmit timeslots and receive timeslots are 0 – 127.

xds_ct_conference_mode() must be called before doing any conferencing and calling and calling this function. This will enable the user's desired conference mode and enable proper error-checking in the library code.

Example

xds_ct_conference(16, 4, 0, 14, 1, 3, 2, 1);

This would use conference handle 4 on board 16 for the conference. Stream 0, timeslot 14 would serve as the output and stream1, timeslot 3 would be the input. The attenuation parameter is set to 2 and the noise threshold is set to 1.

xds_ct_conference_disable

xds_ct_conference_disable(board_number, ct_xmt_stream, ct_xmt_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int ct_xmt_stream;	a value indicating which CT bus stream is serving as an output
int ct_xmt_timeslot;	a value indicating which CT bus timeslot is serving as an output

Applicable boards
H.100/H.110 boards with conferencing resources and the SCSA Enhanced conference board

Purpose
This function will remove the output from the conference bridge.

Message Sent
“CDoooo” where **oooo** is the CT bus stream and timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an illegal output timeslot was used for the board
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will remove a connection from a conference.

Valid values for the transmit streams are 0 – 31. Valid values for the transmit timeslots are 0 – 127.

Example

xds_ct_conference_disable(16, 0, 2);	this would remove stream 0, timeslot 2 from a conference on board 16

xds_ct_conference_dtmf

xds_ct_conference_dtmf(board_number, ct_rcv_stream, ct_rcv_timeslot, time);

unsigned char board_number;	a value indicating which board the command is for
int ct_rcv_stream;	a value indicating which CT bus stream is serving as an input
int ct_rcv_timeslot;	a value indicating which CT bus timeslot is serving as an input
int time;	a value between 1 - 207 indicating the duration of the clamping feature, a value of 0 disables DTMF detection

Applicable Boards

H.100/H.110 boards with conferencing resources and the SCSA Enhanced conference board

Purpose

This function will enable the detection of DTMF tones on a specified CT bus timeslot. The timeslot must be an active input to a conference for detection to be possible. The "clamping" feature can be enabled which will suppress the specified input to the conference for the amount of time specified when a DTMF digit is detected.

Message Sent

"CEiiiidd" where **iiii** is the CT bus stream and timeslot and **dd** is the clamping duration.
"CEiiiiD" if detection/clamping is to be disabled on CT bus timeslot **iiii**.

Returns

This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid value of time was used
ILL_SLOT	an illegal input timeslot was used for the board
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
DTMF detection can be enabled simultaneously on all possible conference inputs. DTMF detection can only be enabled on active inputs to conferences. If the input is removed from the conference using the **xds_ct_conference_disable** function or the conference is dissolved using the **xds_ct_conference_dissolve** function, detection will be disabled.

A "clamping" feature is also provided as part of DTMF function. If this feature is enabled by specifying a time in the range of 1-207, the input to the conference will be suppressed if a DTMF digit is detected. The suppression will begin as soon as a DTMF digit is qualified (approximately 20 msec.) and will continue for the specified time. This time is given in multiples of 20 msec. with a maximum duration of 4.140 seconds. If the time parameter is 0, DTMF detection and clamping will both be disabled.

Valid values for the receive streams are 0 – 31. Valid values for the receive timeslots are 0 – 127.

Example

xds_ct_conference_dtmf(16, 0, 3, 10);	this will enable DTMF detection on stream 0, timeslot 3 with a clamping duration of 200 msec.

xds_ct_conference_monitor

xds_ct_conference_monitor(board_number, handle, ct_xmt_stream, ct_xmt_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int handle;	a value indicating the conference handle to use
int ct_xmt_stream;	a value indicating which CT bus stream is serving as an output
int ct_xmt_timeslot;	a value indicating which CT bus timeslot is serving as an output

Applicable boards
H.100/H.110 boards with conferencing resources and the SCSA Enhanced conference board

Purpose
This function will establish a CT bus output from the conference specified by the handle. No input will be established.

Message Sent
"CMhhoooo" where **oooo** is the CT bus output stream and timeslot and **hh** is the conference handle.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an illegal input and/or output timeslot was used for the board
ILL_HANDLE	an invalid conference handle was chosen for the conference
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
A conference monitor can be treated as a conference party with a silent input. As with a full conference party, a Conference Control Address and conference port is used. A conference monitor may be disabled by using the **xds_ct_conference_disolve** or **xds_ct_conference_disable** functions. If the **xds_ct_conference_monitor** function is called for a conference port that is in use, the input from that port will be removed from the conference. Similarly, if an **xds_ct_conference** function is called for a port that is monitoring, the port will become a full party to the conference.

Valid values for the transmit streams are 0 – 31. Valid values for the transmit timeslots are 0 – 127.

xds_ct_conference_mode() must be called before doing any conferencing and calling and calling this function. This will enable the user's desired conference mode and enable proper error-checking in the library code.

Example

xds_ct_conference_monitor(16, 2, 0, 3);	this will monitor the conference with handle 3 using stream 0, timeslot 2 on board 16.

xds_ct_mix_remove

xds_ct_mix_remove(board_number, handle, ct_rcv_stream, ct_rcv_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int handle;	a value indicating the conference handle to use
int ct_rcv_stream;	a value indicating which CT bus stream is serving as an input
int ct_rcv_timeslot;	a value indicating which CT bus timeslot is serving as an input

Applicable boards
H.100/H.110 boards with conferencing resources and the SCSA Enhanced conference board

Purpose
This function is used to remove an input from a “mix”. It will remove only the input from the mix with the specified handle. Any other mixes the timeslot may be an input to will not be affected.

Message Sent
“CXhhiiii” where **hh** is the mix handle and **iiii** is the CT bus input stream and timeslot

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an illegal input and/or output timeslot was used for the board
ILL_HANDLE	an invalid conference handle was chosen for the conference
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The **xds_ct_mix_remove** function is used to remove inputs from mixes created using the **xds_ct_mix_input** function. See the description of that function for details of how a mix works.

xds_ct_conference_mode() must be called before doing any conferencing and calling and calling this function. This will enable the user’s desired conference mode and enable proper error-checking in the library code.

Example

xds_ct_mix_remove(16, 14, 4);	this will remove timeslot 14 from the mix using handle 4 on board 16.

xds_ct_mix_input

xds_ct_mix_input(board_number, handle, ct_rcv_stream, ct_rcv_timeslot, atten, threshold);

unsigned char board_number;	a value indicating which board the command is for
int handle;	a value indicating the conference handle to use
int ct_rcv_stream;	a value indicating which CT bus stream is serving as an input
int ct_rcv_timeslot;	a value indicating which CT bus timeslot is serving as an input
char atten;	a value between 0-7 that controls the input and output attenuation of the conference on the specified timeslots
char threshold;	a value between 0-3 specifying a noise threshold below which the input will not be added to a conference

Applicable boards
H.100/H.110 boards with conferencing resources and the SCSA Enhanced conference board

Purpose
This function adds an input to a conference “mix”. Using this function, a CT bus timeslot can serve as an input to several different mix outputs, each of which is a combination of a different set of inputs.

Message Sent
“CIhhiiiiat” where **iiii** is the CT bus input stream and timeslot, **hh** is the conference handle being used for the mix, **a** is the attenuation parameter and **t** is the noise threshold parameter.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an illegal input and/or output timeslot was used for the board
ILL_HANDLE	an invalid conference handle was chosen for the conference
ILL_ATTEN	an invalid attenuation value was chosen for the conference
ILL_THRESHOLD	an invalid threshold value was chosen for the conference
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The conference facilities can be used to create a set of mixes where each mix consists of a different set of inputs, but where each input timeslot can serve as an input to one or more different mixes. Each mix is independent of the other mixes and inputs can be added or deleted at any time.

A mix output is established by using the **xds_ct_conference** or **xds_ct_conference_monitor** functions to create a conference output. Inputs are then added to the mix by using the **xds_ct_mix_input** function. Inputs may be removed from a mix by using the **xds_ct_mix_clear** function. Removing an input from one mix will not remove it from any other mixes it is a party to.

The **xds_ct_conference_dissolve** will disable the mix outputs and any inputs to the mix. It will not affect any inputs that may also be part of other mixes. Each mix input to each mix requires a separate Conference Control Address (CCA).

xds_ct_conference_mode() must be called before doing any conferencing and calling and calling this function. This will enable the user's desired conference mode and enable proper error-checking in the library code.

Example

xds_ct_mix_input(16, 14, 1, 4, 1, 0);	this adds stream 1, timeslot 4 on the CT bus as an input to the mix using handle 14. The attenuation parameter is 1 and the noise threshold parameter is 0.

xds_ct_conference_mode

xds_ct_conference_mode(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a character indicating whether conferencing should be enabled 'E', enhanced enabled '2', and disabled 'D'

Applicable boards
H.100/H.110 boards with conferencing resources and the SCSA Enhanced conference board

Purpose
This function is used to enable or disable conferencing on the board.

Message Sent
"SKm" where **m** is the conferencing mode

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid conference mode was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
If conferencing is not enabled, other conference commands may not be successfully sent to the board. On the 256 port H.100 conference board, the user may enable 128 ports or 256 ports of conferencing. Mode 'E' enables 128 ports, and '2' enables 256 ports.

Example

xds_ct_conference_mode(16, 'E');	this enables conferencing on board 16

xds_ct_conference_dissolve

xds_ct_conference_dissolve(board_number, handle);

unsigned char board_number;	a value indicating which board the command is for
int handle;	a value indicating the conference handle to use

Applicable boards
All XDS boards with conferencing

Purpose
This function dissolves an entire conference. All inputs and outputs associated with the specified handle are disabled.

Message Sent
"CUhh" where **hh** is the handle number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_HANDLE	an invalid conference handle was chosen for the conference
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function affects all parties to a conference. All inputs and outputs associated with the handle are disabled and the "CCA's" are returned to the available pool. This function does not disable an output established using the **xds_ct_conference_monitor** function.

xds_ct_conference_mode() must be called before doing any conferencing and calling and calling this function. This will enable the user's desired conference mode and enable proper error-checking in the library code.

Example

xds_ct_conference_dissolve(1, 3);	this will dissolve the conference with a handle of 3.

xds_ct_conference_param

xds_ct_conference_param(board_number, handle, atten, threshold);

unsigned char board_number;	a value indicating which board the command is for
int handle;	a value indicating the conference handle to use
char atten;	a value between 0-7 that controls the input and output attenuation of the conference on the specified timeslots
char threshold;	a value between 0-3 specifying a noise threshold below which the input will not be added to a conference

Applicable boards
All XDS boards with conferencing

Purpose
This function is used to set the attenuation and noise threshold parameters of all parties to a conference at one time. This may be desirable when adding or deleting parties to a conference to optimize audio performance.

Message Sent

"SAhhat"	where **hh** is the conference handle, **a** is the attenuation parameter, and **t** is the noise threshold parameter.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_HANDLE	an invalid conference handle was chosen for the conference
ILL_ATTEN	an invalid attenuation value was chosen for the conference
ILL_THRESHOLD	an invalid threshold value was chosen for the conference
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
As parties are added to a conference, the noise levels and feedback may increase to unacceptable levels. This affect can be reduced by adding attenuation to the input and output levels. The **xds_ct_conference_param** function may be used to globally control the attenuation parameters for a single conference. The amount of attenuation needed, and the number of parties to the conference for which additional attenuation should be added, are dependent on the specific characteristics of the inputs, and will have to be determined for each application. The **xds_ct_conference_param** function affects all inputs and outputs associated with a particular conference handle. It will also affect inputs to mixes established using the **xds_ct_mix_input** function.

xds_ct_conference_mode() must be called before doing any conferencing and calling and calling this function. This will enable the user's desired conference mode and enable proper error-checking in the library code.

Example

xds_ct_conference_param(16, 4, 3, 0);	this will set the attenuation parameter to 3 and the noise threshold parameter to 0 for the conference with handle 4 on board 16.

xds_ct_conference_energy_det

xds_ct_conference_energy_det(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a value of ‘E’ if detection is to be enabled or ‘D’ if detection is to be disabled

Applicable Boards
All XDS boards with conferencing

Purpose
This function will enable or disable the energy detection feature. This feature acts globally on all active conference inputs. The energy levels are reported in a table, which consists of one entry for each CT bus timeslot.

Message Sent
“SDD” if energy detection is to be disabled
“SDE” if energy detection is to be enabled

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid detection mode was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Energy detection can be enabled simultaneously on all possible conference inputs. Energy detection can only be enabled on active inputs to conferences. If the input is removed from the conference using the **xds_ct_conference_disconnect** function or the conference is dissolved using the **xds_ct_conference_dissolve** function, detection will be disabled. The results of energy detection are reported in an array by using the **xds_query_energy_table** function.

Example

xds_ct_conference_energy_det(16, ‘E’);	this will enable energy detection on board 16

xds_ct_query_energy_table

xds_ct_query_energy_table(board_number, &energy);

unsigned char board_number;	a value indicating which board the command is for
unsigned char *array;	a pointer to an array to place the energy values in

Applicable Boards
All XDS boards with **enhanced** conferencing

Purpose
If energy detection is enabled, this function will fill an array with the energy levels of active conference inputs. The table is arranged by CT bus timeslots.

Message Sent
none

Returns
This function will return the following values:

(-1)	the energy table flag was unable to be reset
SUCCESS	the energy table flag has been updated
NO_UPDATE	the energy table has no updated information for user / application
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Energy levels are reported by the board in a table in dual-ported memory, one for each CT bus timeslot. A flag is used to indicate that the table has been updated. A value of 1 indicates an update. This function checks the flag, and if it is 0x1 it transfers the contents of the table to the array and sets the flag to 0x0. If the flag is 0, the function returns with a NO_UPDATE return code. The latter indicates that either energy detection has not been enabled using the x**ds_ct_conference_energy_det** function, or not enough time has been allowed since the last attempt to read the table.

The energy levels reported will be in the range 0x0 to 0x1F where each step corresponds to 3 dB. If a CT bus timeslot is not an input to a conference, the energy value for that timeslot will be 0.

Example

xds_query_energy_table(1, &energy);	this will transfer the energy level values to the *energy* array

SCSA PEB Conference
Board Functions

This page was intentionally left blank.

xds_peb_conf_conference

xds_peb_conf_conference(board_number, handle, output_bus, output_stream, output_timeslot, input_bus, input_stream, input_timeslot, atten, threshold)

unsigned char board_number;	a value indicating which board the command is for
char handle;	a value between 1 and 42 indicating which conference bridge is to be used by the function
char output_bus;	a value of either ‘P’ for PEB or ‘S’ for SCbus as the output
int output_stream;	a value between 0-3h for the PEB bus, or 0-Fh for the Scbus indicating which stream is serving as an output
int output_timeslot;	a value between 0-17h (1.544 PEB), 0-1Fh (2.048 PEB), 0-1Fh (2.048 SCbus), or 0-3Fh (SCbus) indicating which timeslot is serving as an output
char input_bus;	a value of either ‘P’ for PEB or ‘S’ for SCbus as the input
int input_stream;	a value between 0-3h for the PEB bus, or 0-Fh for the Scbus indicating which stream is serving as an input
int input_timeslot;	a value between 0-17h (1.544 PEB), 0-1Fh (2.048 PEB), 0-1Fh (2.048 SCbus), or 0-3Fh (SCbus) indicating which timeslot is serving as an input
char atten;	a value between 0-7 that controls the input and output attenuation of the conference on the specified timeslots
char threshold;	a value between 0-3 specifying a noise threshold below which the input will not be added to a conference

Applicable boards
XDS SCSA PEB Conference Board

Purpose
This function adds a bus connection to the conference with the specified handle. The connection consists of a pair of bus timeslots, one for an input and one for an output. The attenuation and noise threshold parameters used by the conferencing hardware are specified.

Message Sent
“CAhhooooiiiiat” where **hh** is the conference handle, **oooo** is the output bus, stream, and timeslot, **iiii** is the input bus, stream, and timeslot, **a** is the attenuation parameter and **t** is the noise threshold parameter.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ATTEN	an invalid conference attenuation value was chosen
ILL_THRESHOLD	an invalid conference threshold value was chosen
ILL_STREAM	an invalid stream was selected
ILL_HANDLE	an invalid conference handle was chosen
ILL_SLOT	an invalid timeslot was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The conference mechanism is the same as used for conferencing SCbus connections. A bus connection can be a party to a conference with SCbus connections. The same pool of CCA's are used for both buses. Only 32 (24 if the secondary bus is operating in the 1.544 PEB mode) secondary bus inputs and outputs are available for all uses including conferencing.

Example
```
xds_peb_conf_conference(1, 42, ‘S’, 0, 1, ‘S’, 0, 3, 7, 3);
```

this example adds the SCbus bus connection using output stream 0, timeslot 1 and input stream 0, timeslot 3 to the conference with handle 42 on board 1. The attenuation parameter is 7 and the noise threshold parameter is 3.

xds_peb_conf_monitor

xds_peb_conf_monitor(board_number, handle, output_bus, output_stream, output_timeslot)

unsigned char board_number;	a value indicating which board the command is for
char handle;	a value between 1 and 42 indicating which conference bridge is to be used by the function
char output_bus;	a value of either ‘P’ for PEB or ‘S’ for SCbus as the output
int output_stream;	a value between 0-3h for the PEB bus, or 0-Fh for the Scbus indicating which stream is serving as an output
int output_timeslot;	a value between 0-17h (1.544 PEB), 0-1Fh (2.048 PEB), 0-1Fh (2.048 SCbus), or 0-3Fh (SCbus) indicating which timeslot is serving as an output

Applicable boards
XDS SCSA PEB Conference Board

Purpose
This function will create an output on the selected bus from the conference with the specified handle. No input to the conference is created.

Message Sent
“CMhhoooo” where **oooo** is the monitor output bus, stream, and timeslot and **hh** is the conference handle.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_STREAM	an invalid stream was selected
ILL_HANDLE	an invalid conference handle was chosen
ILL_SLOT	an invalid timeslot was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
A timeslot on the output bus can be used to monitor a conference involving connections on the either us. Only 32 outputs to the secondary bus are available (24 if the secondary bus is operating in the 1.544 MHz. PEB mode).

The output from the conference established by this function matches one of the outputs to full participants of the conference, usually the first party added. As such, the level of signal input from that party will be reduced compared to the other parties. This is due to the way the conference hardware adds inputs to create different outputs for each party. This effect is particularly noticeable when the inputs to the conference are coming from T1, E1, or 4-wire E&M circuits.

If an attempt is made to monitor a conference handle when no outputs from that conference have been established, the monitor function will not create an output. A monitor audio path will not be disabled when a conference is dissolved using the **xds_peb_conf_dissolve** function. It is necessary to disable the monitor output using the **xds_peb_conf_disconnect** function.

Example
xds_peb_conf_monitor(1, 42, ‘S’, 0, 1);

this will create a monitor output from the conference with handle 42 on stream 0, timeslot 2 of the SCbus

xds_peb_conf_dissolve

xds_peb_conf_dissolve(board_number, handle)

unsigned char board_number;	a value indicating which board the command is for
char handle;	a value between 1 and 42 indicating which conference bridge is to be used by the function

Applicable boards
XDS SCSA PEB Conference Board

Purpose
This function dissolves a conference.

Message Sent
"CUhh" where **hh** is the conference handle.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_HANDLE	an invalid conference handle was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function dissolves a conference, but does not disconnect the switching path that was created.

Example
xds_peb_conf_dissolve(1, 42);

this example dissolves conference handle 43 on board 1.

xds_peb_conf_mix_clear

xds_peb_conf_mix_clear(board_number, handle, input_bus, input_stream, input_timeslot)

unsigned char board_number;	a value indicating which board the command is for
char handle;	a value between 1 and 42 indicating which conference bridge is to be used by the function
char input_bus;	a value of either 'P' for PEB or 'S' for SCbus as the input
int input_stream;	a value between 0-3h for the PEB bus, or 0-Fh for the Scbus indicating which stream is serving as an input
int input_timeslot;	a value between 0-17h (1.544 PEB), 0-1Fh (2.048 PEB), 0-1Fh (2.048 SCbus), or 0-3Fh (SCbus) indicating which timeslot is serving as an input

Applicable boards
XDS SCSA PEB Conference Board

Purpose
This function is used to remove an input bus, stream, and timeslot from a "mix". It will remove the input only from the handle specified. Any other mixes the timeslot may be an input to will not be affected.

Message Sent
"CXhhiiii" where **hh** is the mix handle and **iiii** is the input bus, stream, and timeslot

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_STREAM	an invalid stream was selected
ILL_HANDLE	an invalid conference handle was chosen
ILL_SLOT	an invalid timeslot was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to remove inputs from mixes created using the **xds_peb_conf_mix_input** function.

Example
xds_peb_conf_mix_clear(1, 42, 'S', 0, 4);

this removes the SCbus bus input on stream 0, timeslot 4 from the mix with handle 42 on board 1.

xds_peb_conf_mix_input

xds_peb_conf_mix_input(board_number, handle, input_bus, input_stream, input_timeslot, atten, threshold)

unsigned char board_number;	a value indicating which board the command is for
char handle;	a value between 1 and 42 indicating which conference bridge is to be used by the function
char input_bus;	a value of either 'P' for PEB or 'S' for SCbus as the input
int input_stream;	a value between 0-3h for the PEB bus, or 0-Fh for the Scbus indicating which stream is serving as an input
int input_timeslot;	a value between 0-17h (1.544 PEB), 0-1Fh (2.048 PEB), 0-1Fh (2.048 SCbus), or 0-3Fh (SCbus) indicating which timeslot is serving as an input
char atten;	a value between 0-7 that controls the input and output attenuation of the conference on the specified timeslots
char threshold;	a value between 0-3 specifying a noise threshold below which the input will not be added to a conference

Applicable boards
XDS SCSA PEB Conference Board

Purpose
This function adds an input from the secondary bus to a conference "Mix". Using this function, an secondary bus timeslot can serve as an input to several different mix outputs, each of which is a combination of a different set of inputs.

Message Sent
"CIhhiiiiat" where **hh** is the conference handle, **iiii** is the input bus, stream, and timeslot, **a** is the attenuation parameter, and **t** is the noise threshold parameter.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ATTEN	an invalid conference attenuation value was chosen
ILL_THRESHOLD	an invalid conference threshold value was chosen
ILL_STREAM	an invalid stream was selected
ILL_HANDLE	an invalid conference handle was chosen
ILL_SLOT	an invalid timeslot was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The conference facilities on the XDS Conference Board can be used to create a set of mixes where each mix consists of a different set of inputs, but where each input timeslot can serve as an input to one or more different mixes. Each mix is independent of the other mixes and inputs can be added or deleted at any time.

A mix is established by first using the **xds_peb_conf_conference** function to create a conference output. Inputs are then added to the mix by using the **xds_peb_conf_mix_input** function**.** Inputs from the bus may be removed from a mix by using the **xds_peb_conf_mix_clear** function. Removing an input from one mix will not remove it from any other mixes it is a party to.

The **xds_peb_conf_dissolve** will disable the mix outputs and any inputs to the mix on either bus. It will not affect any inputs that may also be part of other mixes. Each mix input to each mix requires a separate Conference Control Address (CCA). Each group of 21 handles has 64 CCA's available to it. If the total number of CCA's used exceeds 64, an error will occur and the board will send a message of type 6 sub-type 2.

Example

xds_peb_conf_mix_input(1, 42, 'S', 0, 5, 0, 3);	this adds stream 0, timeslot 5 on the SCbus as an input to the mix using handle 42. The attenuation parameter is 0 and the noise threshold parameter is 3.

xds_peb_conf_connect

xds_peb_conf_connect(board_number, output_bus, output_stream, output_timeslot, input_bus, input_stream, input_timeslot);

unsigned char board_number;	a value indicating which board the command is for
char output_bus;	a value of either 'P' for PEB or 'S' for SCbus as the output
int output_stream;	a value between 0-3h for the PEB bus, or 0-Fh for the Scbus indicating which stream is serving as an output
int output_timeslot;	a value between 0-17h (1.544 PEB), 0-1Fh (2.048 PEB), 0-1Fh (2.048 SCbus), or 0-3Fh (SCbus) indicating which timeslot is serving as an output
char input_bus;	a value of either 'P' for PEB or 'S' for SCbus as the input
int input_stream;	a value between 0-3h for the PEB bus, or 0-Fh for the Scbus indicating which stream is serving as an input
int input_timeslot;	a value between 0-17h (1.544 PEB), 0-1Fh (2.048 PEB), 0-1Fh (2.048 SCbus), or 0-3Fh (SCbus) indicating which timeslot is serving as an input

Applicable boards
XDS SCSA PEB Conference Board

Purpose
This function creates a one-way audio path from one bus to another on the PEB conference board. Up to 32 connections can be made.

Message Sent
"CCooooiiii" where the **oooo** is the output bus, stream, and timeslot and **iiii** is the input bus, stream, and timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_STREAM	an invalid stream was selected
ILL_SLOT	an invalid timeslot was selected
ILL_MODE	an invalid bus was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to create a one-way audio connection from one bus to the other. A bi-directional connection may be made by calling this function twice with the opposite parameters for input and output each time.

Example
xds_peb_conf_connect(1, ‘S’, 0, 1, ‘P’, 0, 2);

this creates an audio path between stream 0, timeslot 1 on the SCbus to stream 0, timeslot 2 on the PEB bus on board 1.

xds_peb_conf_connect(1, ‘P’, 0, 2, ‘S’, 0, 1);

this creates an audio path between stream 0, timeslot 2 on the PEB bus to stream 0, timeslot 1 on the SCbus on board 1.

xds_peb_conf_pattern

xds_peb_conf_pattern(board_number, bus, stream, timeslot, pattern);

unsigned char board_number;	a value indicating which board the command is for
char bus;	a value of either ‘P’ for PEB or ‘S’ for SCbus
int stream;	a value between 0-3h for the PEB bus, or 0-Fh for the Scbus indicating which stream is serving as an output
int timeslot;	a value between 0-17h (1.544 PEB), 0-1Fh (2.048 PEB), 0-1Fh (2.048 SCbus), or 0-3Fh (SCbus) indicating which timeslot is serving as an output
int pattern;	a value between 0x0 and 0xFF to be output to the bus chosen by user

Applicable boards
XDS SCSA PEB Conference Board

Purpose
This function will output a fixed pattern value on the specified timeslot on a selected bus on the conference board. This may be done for diagnostic purposes, or to place “silence” or some other pattern on that particular timeslot.

Message Sent
“CPooooppˮ where the **oooo** is the output bus, stream, and timeslot, and **pp** is the pattern value.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PATTERN	an pattern value was selected
ILL_STREAM	an invalid stream was selected
ILL_SLOT	an invalid timeslot was selected
ILL_MODE	an invalid bus was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will cause the pattern value to be output to the selected bus, stream, and timeslot. Placing a pattern on a timeslot may be used as a diagnostic tool to check bus integrity. It may also be used to place a "silence" pattern of 0xFF on the desired bus.

Example

xds_peb_conf_pattern(1, 'P', 3, 0x1F, 0x45);	this outputs the pattern 0x45 on stream 3, timeslot 0x1F of the PEB bus on board 1.
xds_peb_conf_pattern(1, 'S', 0xF, 0x3F, 0xFF);	this outputs the silence pattern (0xFF) on stream 0xF, timeslot 0x3F of the SCSA bus on board 1.

xds_peb_conf_disconnect

xds_peb_conf_disconnect(board_number, bus, stream, timeslot, pattern);

unsigned char board_number;	a value indicating which board the command is for
char bus;	a value of either 'P' for PEB or 'S' for SCbus
int stream;	a value between 0-3h for the PEB bus, or 0-Fh for the Scbus indicating which stream is serving as an output
int timeslot;	a value between 0-17h (1.544 PEB), 0-1Fh (2.048 PEB), 0-1Fh (2.048 SCbus), or 0-3Fh (SCbus) indicating which timeslot is serving as an output
int pattern;	a value between 0x0 and 0xFF to be output to the bus chosen by user

Applicable boards
XDS SCSA PEB Conference Board

Purpose
This function will disconnect an output bus, stream, and timeslot.

Message Sent
"CDoooo" where the **oooo** is the output bus, stream, and timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_STREAM	an invalid stream was selected
ILL_SLOT	an invalid timeslot was selected
ILL_MODE	an invalid bus was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will disable any connection or conference output to the given stream and timeslot.

Example

xds_peb_conf_disconnect(1, 'P', 3, 0x1F);	this disconnects stream 3, timeslot 0x1F of the PEB bus on board 1.

xds_peb_conf_set_sec_clock

xds_peb_conf_set_sec_clock(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a value between 0x0 and 0xA specifying which clock mode the secondary bus should be set to

Applicable boards
XDS SCSA PEB Conference Board

Purpose
This function is used to set the clock mode of the secondary bus connected to the conference board. This clock mode can be set independently of the mode of the primary bus.

Message Sent
"SC1x" where **x** is the clock mode.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid clock mode was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The secondary bus can be set to run in a variety of clock modes independent of the primary SCbus. In all clock modes, connections are possible between the two buses. The possible clock modes are:

0x0	no clock to the bus
0x1	4.096 MHz SCbus slave
0x2	4.096 MHz SCbus master
0x3	2.048 MHz SCbus slave
0x4	2.048 MHz SCbus master
0x5	2.048 MHz PEB network module
0x6	2.048 MHz PEB network extension module
0x7	2.048 MHz PEB resource module
0x8	1.544 MHz PEB network module
0x9	1.544 MHz PEB network extension module
0xA	1.544 MHz PEB resource module

Example
```
xds_peb_conf_set_sec_clock(1, 5);
```
this sets the secondary bus clock mode on board 1 to 5, a 2.048 PEB network module.

xds_peb_conf_set_sec_nxtclk

xds_peb_conf_set_sec_nxtclk(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a value between 0x0 and 0x2 specifying which next clock mode the secondary bus should be set to

Applicable boards
XDS SCSA PEB Conference Board

Purpose
This function is used to set the next clock mode of the secondary bus connected to the conference board. Using this function, the board can be set to automatically assume the role of master clock for the secondary bus if it detects a clock failure on the secondary bus. The clock mode on the primary bus is not affected.

Message Sent
"SN1x" where **x** is the next clock mode**.**

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid clock mode was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
When operating in an SCbus clock mode, the board has the capability of being able to assume the role of master clock source for the secondary bus if it detects a failure of the clock on that bus. This is done without intervention from the application. When this is done, the board will send a message of type 3, subtype 7. This capability is not available when the secondary bus is operating in a PEB mode.

The allowed next clock modes are:

0 - will not become next clock
1 - will become a 4.096 MHz. SCbus clock master on the secondary bus
2 - will become a 2.048 MHz. SCbus clock master on the secondary bus

The board has the capability of becoming the Next Clock on the primary SCbus. However, next clock capability is not part of the 4.1 SC specification and is not supported by the library.

Example
```
xds_peb_conf_set_sec_nxtclk(2, 1);
```

this sets board 2 to become the next 4.096 MHz. clock source on the secondary bus in the event of a clock failure.

This page was intentionally left blank.

Music On Hold
Audio Port Functions

This page was intentionally left blank.

xds_audio_disable_music_rcv

xds_audio_disable_music_rcv(board_number);

unsigned char board_number;	a value indicating which board the command is for

Applicable boards
Infinity Series boards that have a music on hold audio port

Purpose
This function is used to disable the music on hold audio port output from the CT bus.

Message Sent
"ARX"

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Infinity Series MC3 and Conference boards are equipped with an audio port that can operate in either the transmit or receive direction, i.e. output from the analog port to a CT bus timeslot or output from the CT bus to the analog port. This function disables output from the port to the bus.

Example

xds_audio_disable_music_rcv(16, 1);	this will disable the audio port receive on board 16.

xds_audio_disable_music_xmt

xds_audio_disable_music_xmt(board_number);

unsigned char board_number;	a value indicating which board the command is for

Applicable boards
Infinity Series boards that have a music on hold audio port

Purpose
This function is used to disable the music on hold audio port output to the CT bus.

Message Sent
"AD"

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Infinity Series MC3 and Conference boards are equipped with an audio port that can operate in either the transmit or receive direction, i.e. output from the analog port to a CT bus timeslot or output from the CT bus to the analog port. This function disables output from the bus to the port.

Example

xds_audio_disable_music_xmt(16);	this will disable the audio port transmit on board 16.

xds_audio_enable_music_rcv

xds_audio_enable_music_rcv(board_number, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int timeslot;	a value indicating which timeslot on the CT bus is serving as an input to the audio port

Applicable boards
Infinity Series boards that have a music on hold audio port

Purpose
This function is used to enable the input from the CT bus to the music on hold audio port.

Message Sent
"ARssst" where **sstt** is the CT bus stream and timeslot

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot value was called for the board being used
WRONG_BOARD	an non-PCI based board was called
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Infinity Series MC3 and Conference boards are equipped with an audio port that can operate in either the transmit or receive direction, i.e. output from the analog port to an CT bus timeslot or output from the CT bus to the analog port. This function enables an input from the CT bus to the port.

Example

xds_audio_enable_music_rcv(16, 27);	this will enable input from timeslot 27 to the audio port on board 16.

xds_audio_enable_music_xmt

xds_audio_enable_music_xmt(board_number, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int timeslot;	a value indicating which timeslot on the CT bus the audio port is outputting to

Applicable boards
Infinity Series boards that have a music on hold audio port

Purpose
This function is used to enables output from the music on hold audio port to the CT bus.

Message Sent
"AEsstt" where **sstt** is the CT bus stream and timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot value was called for the board being used
WRONG_BOARD	an non-PCI based board was called
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Infinity Series MC3 and Conference boards are equipped with an audio port that can operate in either the transmit or receive direction, i.e. output from the analog port to an CT bus timeslot or output from the CT bus to the analog port. This function enables output from the port to the bus.

Example

xds_audio_enable_music_xmt(16);	this will enable the audio port transmit on board 16.

xds_audio_music_atten

xds_audio_music_atten(board_number, xmt_attn, rcv_attn);

unsigned char board_number;	a value indicating which board the command is for
int xmt_atten;	a value between 0x00 and 0xFF indicating the transmit attenuation in .1 dB steps
int rcv_atten;	a value between 0x00 and 0xFF indicating the transmit attenuation in .1 dB steps

Applicable boards
Infinity Series boards that have a music on hold audio port

Purpose
This function is used to control the levels of the music on hold audio port.

Message Sent
"AGttrr" where **tt** is the transmit attenuation and **rr** is the receive attenuation.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Infinity Series MC3 and Conference boards are equipped with an audio port that can operate in either the transmit or receive direction, i.e. output from the analog port to an CT bus timeslot or output from the CT bus to the analog port. This function controls the audio levels in the transmit and receive directions. The amount of attenuation is controllable in .1 dB steps over a range of 0 dB to -25.5 dB.

Example

xds_audio_music_atten(16, 30, 0);	this will set the audio port transmit attenuation to 3 dB and the receive attenuation to 0 dB on board 16.

xds_audio_music_enable

xds_audio_music_enable(board_number, enable);

unsigned char board_number;	a value indicating which board the command is for
char enable;	a value of either ‘0’ or ‘1’ indicating the enable mode

Applicable boards
Infinity Series boards that have a music on hold audio port

Purpose
This function is the MVIP-compatibility mode function used to control the enabling and disabling of the music on hold audio port.

Message Sent
“MAx” where **x** is the enable mode to be used.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Infinity Series MC3 and Conference boards are equipped with an audio port that can be enable or disabled using this function.

Example

xds_audio_music_atten(16, 1);	this will enable the audio port on board 16

T1/E1 Common Switching and Layer 3 Board Functions

This page was intentionally left blank.

xds_t1e1_blue_alarm

int xds_t1e1_blue_alarm(board_number, b_channel, mode);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char mode;	a character indicating which mode to be used

Applicable Boards
T1 / E1 boards

Purpose
This function can be used to set or clear the blue alarm event on a given channel.

Message Sent
"ABxxc" where **xx** is the B-channel and **c** is the alarm mode

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The modes available are 'S' to set the event and 'C' to clear the event.

Example

xds_t1e1_blue_alarm (16, 0, 'S');	sets a blue alarm event on channel 0

xds_t1e1_yellow_alarm

int xds_t1e1_yellow_alarm(board_number, b_channel, mode);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char mode;	a character indicating which mode to be used

Applicable Boards
T1 / E1 boards

Purpose
This function can be used to set or clear the yellow alarm event on a given channel.

Message Sent
"AYxxc" where **xx** is the B-channel and **c** is the alarm mode

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The modes available are 'S' to set the event and 'C' to clear the event.

Example

xds_t1e1_yellow_alarm (16, 0, 'S');	sets a yellow alarm event on channel 0

xds_t1e1_listen

int xds_t1e1_listen(board_number, b_channel, steam, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int stream;	the selected stream to listen to
int timeslot;	the selected timeslot to listen to

Applicable Boards
T1 / E1 boards

Purpose
This function is used to set a B-channel to listen to a stream and timeslot

Message Sent
"CAxxsstt" where **xx** is the B-channel, **ss** is the stream, and **tt** is the timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
ILL_SLOT	an invalid stream or timeslot was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The valid range of streams is from 0 to 31, while the valid range of timeslots is 0 to 127.

Example

xds_t1e1_listen(16, 11, 0, 3);	sets B-channel 11, on board 16, to listen to stream 0 timeslot 3

xds_t1e1_connect

int xds_t1e1_connect(board_number, b_channel, output_steam, output_timeslot input_steam, input_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int output_stream;	the output stream to connect
int output_timeslot;	the output timeslot to connect
int input_stream;	the input stream to connect
int input_timeslot;	the input timeslot to connect

Applicable Boards
T1 / E1 boards

Purpose
This function is used to connect an input stream and timeslot with an output stream and timeslot to a B-channel.

Message Sent
"CCxxssttaabb" where **xx** is the B-channel, **ss** is the output stream, **tt** is the out put timeslot, **aa** is the input stream, and **bb** is the input timeslot

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
ILL_SLOT	an invalid stream or timeslot was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The valid range of streams is from 0 to 31, while the valid range of timeslots is 0 to 127.

Example

xds_t1e1_connect(16, 11, 0, 3, 10, 50);	connects input stream 0 timeslot 3 and output stream 10 timeslot 50 to B-channel 11, on board 16

xds_t1e1_disconnect

int xds_t1e1_disconnect(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable Boards
T1 / E1 boards

Purpose
This function is used to disconnect a B-channel.

Message Sent
"CDxx" where **xx** is the B-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The layer three equivalent to this function is the xds_t1_l3_disconnect() function.

Example

xds_t1e1_disconnect(16, 11);	disconnects B-channel 11 on board 16

xds_t1e1_hold

int xds_t1e1_hold(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable Boards
T1 / E1 boards

Purpose
This function is used to send a HOLD message to a selected B-channel.

Message Sent
"CHxx" where **xx** is the B-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The layer three equivalent to this function is the xds_t1_l3_hold() function.

Example

xds_t1e1_hold (16, 11);	puts B-channel 11, on board 16, on HOLD

xds_t1e1_detect_dtmf

int xds_t1e1_detect_dtmf(board_number, b_channel, mode);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char mode;	a value indicating the DTMF detection mode to be used

Applicable Boards
T1 / E1 boards

Purpose
This function is used to control the DTMF detection and the mode of it on a selected B-channel.

Message Sent
"CLxx" where **xx** is the B-channel
or
"CLxxo" where **xx** is the B-channel and **o** is the option

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid detection mode was called for the board being used
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The valid modes for DTMF detection are: 0 (no option) to play dial tone, 'F' to turn detection off, 'M' for monitor mode & maintain existing connections, and 'Q' for regular mode & suppress dial tone.

Examples

xds_t1e1_detect_dtmf (16, 11, 0);	detects DTMF on B-channel 11 – play dial tone, on board 16
xds_t1e1_detect_dtmf (16, 13, 'M');	detects DTMF on B-channel 13 – monitor mode, on board 16
xds_t1e1_detect_dtmf (16, 15, 'F');	disable DTMF detection on B-channel 15, board 16

xds_t1e1_give_call_progress

int xds_t1e1_give_call_progress(board_number, b_channel, tone);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char tone;	a value indicating the call progress tone to be used

Applicable Boards
T1 / E1 boards

Purpose
This function is used to give call progress tones on a given B-channel.

Message Sent
"CPxxy" where **xx** is the B-channel and **y** is the call progress tone

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid call progress tone was called for the board being used
ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The valid modes for call progress are:
'0' – dial tone
'1' – reorder
'2' – busy signal
'3' – audible ringback
'4' – digital milliwatt
'5' – silence

Examples

xds_t1e1_give_call_progress(16, 11, '0');	give B-channel 11 a call progress "dial tone" tone on board 16
xds_t1e1_give_call_progress(16, 13, '2');	give B-channel 13 plays a call progress "busy signal" tone on board 16

xds_t1e1_dial_string

int xds_t1e1_dial_string(board_number, b_channel, dial string);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to be used
char *dial_string;	a pointer to a dial string of characters

Applicable Boards
T1 / E1 boards

Purpose
This function is used to send a string of dial string characters to a given B-channel.

Message Sent
"CTxx(ds)" where **xx** is the B-channel to be used and **ds** is the dial string

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The valid tone characters in the dial string are as follows: 0-9, *, #, A-D, U - upper tone (941 Hz.), L - lower tone, (697 Hz.), X - short pause, P - long pause.

Examples

xds_t1e1_dial_string (16, 11, "01234");	send the dial string "01234" to B-channel 11 on board 16
xds_t1e1_dial_string (16, 13, 'U012XL345"');	send the dial string "012" in a 941 Hz upper tone, a short pause, then send "345" in a 697 Hz lower tone to B-channel 11 on board 16

xds_t1e1_transmit

int xds_t1e1_transmit (board_number, b_channel, stream, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to be used
int stream;	a value indicating which stream is to be used
int timeslot;	a value indicating which timeslot is to be used

Applicable Boards
T1 / E1 boards

Purpose
This function is used set the B-channel to transmit on a stream and timeslot.

Message Sent
"CXxxsstt" where **xx** is the B-channel to be used, **ss** is the stream, and **tt** is the timeslot

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel was called for the board being used
ILL_SLOT	an invalid stream or timeslot was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The transmit command is used to setup a B-channel to transmit towards the H.100 bus. There are 32 streams available on each board. On each stream, there are 128 timeslots available.

Examples

xds_t1e1_transmit(16, 11, 2, 1);	would direct timeslot 1 of stream 2 to B-channel 11, on board 16

xds_t1e1_l3_disconnect

xds_t1e1_l3_disconnect(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 DISConnect message for the call currently associated with the specified B-Channel. A DISConnect message is used to notify either the network or the user that a disconnect sequence has been initiated.

Message Sent
"DDxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a DISConnect message to be sent for the call associated with the specified B-channel. This should be done when initiating a disconnect sequence.

Example

xds_t1e1_l3_disconnect(16, 0);	this will send a DISConnect message for B-channel 0 on board 16

xds_t1e1_l3_retrieve

xds_t1e1_l3_retrieve(board_number, b_channel, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *reference;	a pointer to a character string to be used as the call reference number

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 RETrieve message to a B-channel.

Message Sent
"DGxxHrrrr" where **xx** is the B-channel number, and **rrrr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	invalid call reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a RETrieve message to a selected b-channel on a selected board. Valid values for the call reference number are between "0000" and "7FFF".

Example

xds_t1e1_l3_retrieve(16, 0, "0193");	this sends a message (B-channel 0 to board 16 with a call reference # of 0x193)

xds_t1e1_l3_retrieve_ack

xds_t1e1_l3_retrieve_ack(board_number, b_channel, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *reference;	a pointer to a character string to be used as the call reference number (for CI spans)

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 RETrieve ACKnowledge message to a B-channel.

Message Sent
"DGxxArrrr" where **xx** is the B-channel number, and **rrrr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	invalid call reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a RETrieve ACKnowledge message to a selected b-channel on a selected board. Valid values for the call reference number are between "0000" and "7FFF".

Example

xds_t1e1_l3_retrieve_ack(16, 4, "0193");	this sends a message (B-channel 4 to board 16 with a call reference # of 0x193)

xds_t1e1_l3_retrieve_rej

xds_t1e1_l3_retrieve_rej(board_number, b_channel, cause, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value of the disconnect cause
char *reference;	a pointer to a character string to be used as the call reference number (for CI spans)

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 RETrieve REJect message to a B-channel.

Message Sent
"DGxxArrrr" where **xx** is the B-channel number, and **rrrr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	invalid call reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a RETrieve REJect message to a selected b-channel on a selected board. Valid values for the call reference number are between "0000" and "7FFF".

Example

xds_t1e1_l3_retrieve_ack(1, 0x4, 0x19, "0093");	this sends a message (B-channel 4 to board 1 with a cause value of 0x19 and a call reference # of 0x93)

xds_t1e1_l3_hold

xds_t1e1_l3_hold(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 HOLD message to a B-channel.

Message Sent
"DHxx" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD message to a selected b-channel on a selected board.

Example

xds_t1e1_l3_hold(16, 4);	this sends a message (B-channel 4 to board 16)

xds_t1e1_l3_hold_ack

xds_t1e1_l3_hold_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 HOLD ACKnowledge message to a B-channel.

Message Sent
"DHxxA" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD ACKnowledge message to a selected b-channel on a selected board.

Example

xds_t1e1_l3_hold_ack(16, 4);	this sends a message (B-channel 4 to board 16)

xds_t1e1_l3_hold_rej

xds_t1e1_l3_hold_rej(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value of the disconnect cause

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 HOLD REJect message to a B-channel.

Message Sent
“DHxxRcc” where **xx** is the B-channel number, and cc is the cause for the rejection

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a HOLD REJect message to a selected b-channel on a selected board.

Example

xds_t1e1_l3_hold_rej(16, 4, 0x7F);	this sends a message (B-channel 4 to board 16, cause 0x7F)

xds_t1e1_l3_progress

xds_t1e1_l3_progress(board_number, b_channel, cause, progress);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value of the disconnect cause
char progress;	a character indicating the progress indicator if any

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 PROGress message for the call currently associated with the specified B-Channel. A PROGress message is used to notify the caller of interworking with non-ISDNs, a call is routed to an inband tone or announcement, or when a progress delay at the destination is reported.

Message Sent
"DPxxPccp" where **xx** is the B-channel number, **cc** is the cause value, and **p** is the progress indicator

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_PROGRESS	invalid progress indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a PROGress message to be sent for the call associated with the specified B-channel. A PROGress message is sent to report an interworking situation such as the destination is non ISDN, or a situation where the call, though initiated can not be completed, such as the called party is busy. In those cases, an inband signal such as busy tone is usually present and indicated by the progress indicator in the message. An application would send a PROGress message when it realizes that a call can not be completed, or when interworking with the destination switch results in the receipt of a PROGress message from the destination.

The cause code is a value used to indicate the reason for the PROGress message. Examples would be a cause code of 0x11, "User busy", or 0x12, "No user responding". For a list of cause codes and their meaning, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

Valid values for the call progress indicator are:
C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

Example

xds_t1e1_l3_progress(16, 4, 0x11, 'N');	this will send a PROGress message for B-channel 4 on board 16. The cause is 0x11, "User busy", with no progress indicator

xds_t1e1_l3_release

xds_t1e1_l3_release(board_number, b_channel, port_type, cause, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a value indicating the span type (NT or CI)
int cause;	a value of the disconnect cause
char *reference;	a pointer to a character string to be used as the call reference number

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 RELease message for the call currently associated with the specified B-Channel. A RELease message is used to release a call reference and terminate a call in cases where a DISConnect message is inappropriate.

Message Sent
E1 NT Span: "DRxxRcc(rrrr)" where **xx** is the B-channel number, **cc** is the cause value, and **rrrr** is the optional call reference.
T1 CI Span: "DRxxR(cc)(rrrr)" where **xx** is the B-channel number, **cc** is the optional cause value, and **rrrr** is the optional call reference.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	out of range call reference number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a RELease message to the call associated with the specified B-channel. This should be done to release a call and associated call reference under some circumstances. An example is when no TE equipment has responded to a SETUP message. If the call has been allocated a B-channel, a DISConnect should be sent to clear a call (see **xds_t1e1_l3_disconnect**).

The cause code is a value used to indicate the reason for the RELease message. Examples would be a cause code of 0x10, "Normal clearing", or 0x41, "Bearer capability not supported". For a list of cause codes and their meaning, see the XDS Layer 3 Protocol Software Reference Manual or Q.931.

For any "optional" values, use a 0 (NULL value) to pass when not using them.

Valid values for the call reference number are between "0000" and "7FFF".

Example

xds_t1e1_l3_release(16, 4, 0x10, "0193");	this will send a RELease message for B-channel 4 on board 16 with a cause value of 0x10 - "Normal Clearing" and a call reference number of 0x193

xds_t1e1_l3_setup_overlap

xds_t1e1_l3_setup_overlap(board_number, b_channel, bearer_cap, progress, calling_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char bearer_cap;	a character indicating the type of call, i.e. speech or data
char progress;	a character indicating the progress indicator
char *calling_number;	a pointer to an ASCII string with the calling number

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 SETUP message, overlap sending. A SETUP message is used to initiate a call. As the SETUP message takes a different form for NT and TE interface, the presence of some arguments will depend on the port type.

Message Sent
T1 CI / E1 TE spans:
"DSxxbp(#)" where **xx** is the B-channel number, **b** is the bearer capability, **p** is the progress indicator, and the # is the optional calling number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
ILL_ARG	invalid bearer capability value, calling number, or called number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SETUP message to be sent to initiate a call. The board level command takes a different form depending on whether the port is configured as an NT (network termination) or a TE (terminal equipment). The treatment of the directory numbers is different for the two port types. The port type must be specified by the **port_type** argument. Valid values are ‘N’ for an NT interface and ‘T’ for a TE interface.

The bearer capability indicates the type of call being made, i.e. voice or data. There are four valid choices:
A - 3.1 kHz audio, 64 kbps, circuit mode, Mu-Law
D - Unrestricted digital information, 64 kbps, circuit mode
R - Rate adaption from 56 kbps, 64 kbps, circuit mode
S - Speech, 64 kbps, circuit mode, Mu-Law

Valid values for the call progress indicator are:
C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

Note that if there are no interworking problems, the progress indicator should be set to ‘N’.
For TE ports, the called number is the subscriber number of the destination. The calling number element will use the default DN of the B-channel on which the call is being placed. The called number can be up to 15 digits long.

Examples
xds_t1e1_l3_setup_overlap(16, 4, ‘T’, ‘D’, ‘’, “8384194”);

this will send a SETUP message for B-channel 4 on board 16 with a bearer capability of data, and a calling number of 838-4194.

xds_t1e1_l3_query_status

xds_t1e1_l3_query_status(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 status enquiry message to the board. This can be done to prompt a terminal to respond with a status message.

Message Sent
"DXxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to query a B-channel about the call state of a call in a case where a received message indicates there may be a call processing discrepancy. The terminal, if it responds will send a STATUS message. As this response is dependent on the terminal and not on the board, this message will be returned on the receive message queue and will be retrieved with the **xds_msg_receive** function rather than the **xds_query_receive** function. This function is only valid for B-channels that have a call in process.

Example

xds_t1e1_l3_query_status(16, 4)	This will query the status of B-channel 4 on board 16.

xds_t1e1_set_protocol

int xds_t1e1_set_protocol (board_number, b_channel, line_type, direction, protocol, digits);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the span to control
char line_type;	a character indicating the line type to use
char direction;	a character indicating direction to use
char protocol;	a character indicating the optional address digit protocol to use
char digits;	a character indicating number of address digits to be expected on incoming calls

Applicable Boards
T1 / E1 boards

Purpose
This function is used to set the robbed-bit line protocol for each channel.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
ILL_TYPE	an invalid line type was selected
ILL_MODE	an invalid direction was selected
ILL_FEATURE	an invalid address digit protocol was used
ILL_ARG	an out of range number of address digits was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Message Sent
"SPxxtd(pd)" where **xx** is the span (interface), **t** is the line type, the first **d** is the direction mode, **p** is the optional address digit protocol, and the second **d** is the optional number of address digits to be expected

Comments
This function is one of the functions to be used to set up the board before functional use.

The interface number is specified in **xx**.

The **t** parameter specifies the line type, ‘E’ for E&M, ‘G’ for Ground Start, ‘L’ for Loop Start, or ‘N’ for None.

The first **d** parameter specifies the direction, ‘S’ for the FXS or CPE side, and ‘O’ for the FXO or CO side.

If the line type is E&M, an additional address digit protocol may be specified with the p parameter. A value of ‘I’ specifies immediate start and ‘W’ specifies wink start.

The second **d** parameter specifies the number of address digits (1 to 15) to be expected on incoming calls.

For any “optional” values, use a 0 (NULL value) to pass when not using them.

Example

xds_t1e1_set_protocol(16, 0, ‘E’, ‘S’, ‘W’, ‘7’);	sets the B-channel 0 line protocol to E&M, direction to FXS/CPE side, with a wink start address digit protocol, and expect 7 address digits on the incoming call

xds_t1e1_l3_restart

xds_t1e1_l3_restart(board_number, b_channel, option);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char option;	a value indicating whether or not to restart all interfaces

Applicable boards
T1 / E1 boards

Purpose
This function is used to restart all interfaces or a specified interface.

Message Sent
T1 NT / CI spans:
“DXxx(A)” where **xx** is the B-channel number – restart all interfaces
“DXxxC” where **xx** is the B-channel number – restart a specified interface

T1 NT / CI spans:
“DXxx” where **xx** is the B-channel number – restart an interface staring at B-channel **xx**
“DXxx(C)” where **xx** is the B-channel number – restart a specified interface

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to restart all interfaces or a specified interface, depending on the option selected and the board / span type.

Examples
T1:

xds_t1e1_l3_restart(16, 4)	This will restart B-channel 4 on board 16.
xds_t1e1_l3_restart(16, 4, ‘A’)	This will restart interface 4, all interfaces, on board 16.

E1:

xds_t1e1_l3_restart(17, 4)	This will restart interface starting at B-channel 4 on board 17.
xds_t1e1_l3_restart(17, 4, ‘C’)	This will restart B-channel 4 on board 17.

xds_t1e1_l3_include_name

xds_t1e1_l3_include_name(board_number, invoke_id, operation, name);

unsigned char board_number;	a value indicating which board the command is for
int invoke_id;	a value indicating the ID code
char operation;	a value indicating the operation mode
char *name;	a value indicating the name

Applicable boards
T1 / E1 boards

Purpose
This function is used to restart all interfaces or a specified interface.

Message Sent
"D@Nido<name>" where **xx** is the B-channel number – restart all interfaces

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ID	the ID code is out of valid range
ILL_MODE	an invalid operation mode was selected
ILL_ARG	the name value is NULL
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to include a calling or connected name in a SETUP or CONNECT message.

The four operation modes available to this function are selected with the **operation** argument. They are:

‘0’ – calling name
‘1’ – called name
‘2’ – connected name
‘3’ – busy name

Examples
xds_t1e1_l3_include_name(16, 0xFF, ‘3’, “Amtelco”)

This will indicate a busy name of “Amtelco” with an invoke ID code of 0xFF.

xds_t1e1_l3_include_name(16, 0x93, ‘1’, “John Smith”)

This will indicate a called name of “John Smith” with an invoke ID code of 0x93.

xds_t1e1_l3_message

int xds_t1e1_l3_message(board_number, span, sapi, tei, type, l3MsgLen, l3message);

unsigned char board_number;	a value indicating which board the command is for
int span;	a value indicating the span to control
int sapi;	a value between 0x0 and 03F indicating the SAPI to be used
int tei;	a value between 0x0 and 0x7F indicating the TEI to be used
char type;	either 'C' for command or 'R' for response
unsigned short l3MsgLen;	the number of octets in the Layer 3 message
unsigned char *l3message;	a pointer to an array containing the Layer 3 message

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 message using the DLCI specified by the SAPI and TEI specified on the T1/E1 interface specified by the port argument. The message is contained in the array pointed to by *message and has a length in octets of l3MsgLen.

Message Sent
"LCddsstt" where **C** is the message type, **dd** is the span number, **ss** is the SAPI, and **tt** is the TEI number. The Layer 3 message and length is placed in the auxiliary mailbox.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_TYPE	invalid message type
ILL_ARG	the TEI specified is outside the range 0x0 to 0x7E, or the SAPI is outside the range 0x0 to 0x3F.
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a Layer 3 message using the data link specified by the DLCI (Data Link Connection Identifier) composed of the SAPI (Service Access Point Identifier) and TEI (Terminal Endpoint Identifier). The SAPI has a range of 0-63 while the TEI has a range of 0-127. The normal rules for allowed DLCI addresses apply, and the board currently only supports DLCIs of [63, 127] and in the range [0, 0-127] and [16, 0-126]. The Layer 3 message itself consists of 1 to 260 octets. The contents of the message are not restricted to a Q.931 format. The TEI must already be assigned to the interface if it is in the range 0-126. The broadcast TEI of 127 is always available, and will result in an unnumbered information frame being sent. Two types of messages are allowed, commands and responses, specified by a type of ‘C’ or a ‘R’. The normal type should be ‘C’ or command, and Q.931 specifies only commands.

Example
xds_t1e1_l3_message(0, 2, 0, 0x40, ‘C’, 4, cnctmsg);

This will send a Layer 3 message 4 octets in length with a SAPI of 0 and a TEI of 64 on port 2 of board 0.

In this case, cnctmsg is an array of four octets that is a Q.931 connect
message.cnctmsg[4] = {0x8, 0x1, 0x81, 0x7};

xds_t1e1_set_layer

xds_t1e1_set_layer (board_number, layer);

unsigned char board_number;	a value indicating which board the command is for
char *layer;	a pointer to a character array indicating the protocol layer for each BRI interface, the array consists of a character for each interface on the board. Valid characters are ‘2’, ‘3’, ‘A’, ‘D’, ‘E’, or ‘N’.

Applicable boards
T1 / E1 boards

Purpose
This function is used to set the protocol layer for each of the BRI interfaces. The board can provide support at layer 2, layer 3, AT&T Custom, DMS-100, CACH EKTS, or National ISDN 1.

Message Sent
“SL(p*number of ports)” (ISA / H.100 boards) where p is the protocol layer
“SLL(p*number of ports)” (H.110 boards) where p is the protocol layer, lower bank of 16
“SLH(p*number of ports)” (H.110 boards) where p is the protocol layer, upper bank of 16

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	invalid layer type
WRONG_BOARD	a non-BRI board was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The board will provide support at either layer 2 or layer 3. If layer 2 support is chosen, the board will handle all layer 2 functions, but will require the application to compose and interpret the raw Q.931 messages for call control. These messages are sent using the **xds_t1e1_l3_message()** function**.** If layer 3 support is chosen, call control communications between the driver and the board is accomplished through 'D' messages. This is a simpler application interface and does not require the program to have as detailed a knowledge of the Q.931 message structure.

Layer types are as follows:

T1:
2 - Layer 2 support only
3 - Layer 3 support
A - AT&T custom
D - Nortel DMS-100
N - National ISDN 1/2
S - Siemens CoreNet

E1:
2 - basic Layer 2 support
3 - basic Layer 3 support

The default support is layer 2. The protocol layer supported can be saved in EEAROM using the **xds_set_config** function. If this is done, the support layer and other information will automatically be restored on a board reboot. Therefore, this function will only be needed if the configuration changes.

Example

xds_t1e1_set_layer (16, "3322");	this will set board 16 to support layer 3 on the first two six interfaces and layer 2 on the last six.

xds_t1e1_hookflash

xds_t1e1_hookflash(board_number, b_channel, time);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value between 0 and the maximum number of B-channels on the board indicating which port should receive the tone
int time;	a value between indicating the duration in .1 sec. or .01 sec, depending on the mode, of the hookflash

Applicable Boards
T1 / E1 boards

Purpose
This function will send a hook-flash out of the B-channel. The duration of the hook-flash is in.1 second increments by the **time** argument. Hook-flash signals may be used to get the attention of a PBX or other phone system for performing some action such as transferring a call.

Message Sent
“CFxxd” where **xx** is the B-channel number and **d** is the duration in .1 second increments.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
ILL_ARG	an invalid value for the hookflash time was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The range of hook-flashes that may be sent is .1 to 1.5. If current is not flowing in the line circuit, this command will cause a port to hangup. No state change response message is sent for this function.

Example

xds_t1e1_hookflash(16, 2, 1, 7);	will cause a hook-flash of .7 seconds on port 2 of board 16

xds_t1e1_tei_cnct

int xds_t1e1_tei_cnct(board_number, span, tei);

unsigned char board_number;	a value indicating which board the command is for
int span;	the number of the span that indicates the interface
int tei;	a value between 0x0 and 0x7E indicating the TEI of the data link to be established, a value of 0x7F indicates the packet data link

Applicable boards
T1 / E1 boards

Purpose
This function is used to request the establishment of the DLCI specified by a SAPI of 0 and the Terminal Endpoint Identifier (TEI) specified or the packet data link on the BRI interface specified with the port argument.

Message Sent
"TExxtt" where **xx** is the interface number and **tt** is the TEI number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition.

ILL_ARG	the TEI specified is outside the range 0x0 to 0x7F
ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to request that the data link specified by DLCI (Data Link Connection Identifier) address of [0, TEI] be established. If the TEI value is 0x7F, the packet data link with a SAPI of 16 is specified. The data link must be in the TEI assigned state. If successful, the data link will be put into the multi-frame established state.

Example

xds_t1e1_tei_cnct(16, 2, 0x40);	this will request establishment for a TEI of 64 (0x40) on interface 2 of board 16.

xds_t1e1_tei_disc

int xds_t1e1_tei_disc(board_number, span, tei);

unsigned char board_number;	a value indicating which board the command is for
int span;	the number of the span that indicates the interface
int tei;	a value between 0x0 and 0x7E indicating the TEI of the data link to be disconnected, a value of 0x7F indicates the packet data link

Applicable boards
T1 / E1 boards

Purpose
This function is used to request the disconnection of the DLCI specified by a SAPI of 0 and the Terminal Endpoint Identifier (TEI) specified or the packet data link on the BRI interface specified with the port argument.

Message Sent
"TDxxtt" where **xx** is the interface number and **tt** is the TEI number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the TEI specified is outside the range 0x0 to 0x7F
ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to request that the data link specified by DLCI (Data Link Connection Identifier) address of [0, TEI] be disconnected. If the TEI argument is 0x7F, the packet data link with a SAPI of 16 is specified. Disconnection does not cause the removal of the TEI, but does place it in the TEI assigned state. The link may be reconnected by an attempt to send a message or a call to **xds_tei_cnct**.

Example

xds_t1e1_tei_disc(16, 2, 0x40);	this will request a disconnection for a TEI of 64 (0x40) on interface 2 of board 16.

xds_t1e1_l3_connect_ack

xds_t1e1_l3_connect_ack(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
T1 / E1 boards

Purpose
This function is used to send a Layer 3 connect ACKnowledge message to a B-channel.

Message Sent
"DCxxA" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to send a connect ACKnowledge message.

Example

xds_t1e1_l3_connect_ack(16, 0x4);	this sends a reconnect ACKnowledge message to B-channel 4 on board 16

xds_t1e1_hold_not_busy

xds_t1e1_hold_not_busy(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value between 0 and the maximum number of B-channels on the board indicating which B-channel which B-channel to control

Applicable Boards
T1 / E1 boards

Purpose
The purpose of this function is to seize an idle port and place it in the hold state. If a B-channel is not idle, it will not be seized and a "busy" state change will be returned. This function can be used to minimize "glare" or collisions between incoming and outgoing calls.

Message Sent
"CBxx" where **xx** is the B-channel.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

(-1)	the port is busy
ILL_PORT	an interface port was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function may be used for reducing the possibility of "glare" by calling the function before making an outgoing call on a port. If the port is busy because an incoming call has been detected, the function will return a (-1) and the board will return a "SBxx" message. An additional state change message, such as ring detect or off-hook, indicating the incoming call on that port should also be received.

Example

xds_t1e1_hold_not_busy(16, 2);	this will seize port 2 on board 16

xds_t1e1_set_dn

xds_t1e1_set_dn(board_number, b_channel, dn);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	the number of the B-channel that indicates the interface
char *dn;	a pointer to a character array containing a seven digit ASCII string with the directory number

Applicable boards
T1 / E1 boards

Purpose
This function is used to set the default DN or directory number associated with a particular B-channel. This function is only relevant when layer 3 support is selected.

Message Sent
"SDddddddd" where **ddddddd** is the directory number for the B-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	invalid length of / or NULL directory number
ILL_PORT	the port is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
If layer 3 support is chosen using the **xds_t1e1_set_layer**, it is necessary to define a default directory number for each B-channel. On a port defined as a network termination or NT, the number is used as the called party number in SETUP messages. If the port is defined as a TE, the number is the calling party in SETUP messages. It is possible for both B-channels on an interface to have the same directory number.

Example

xds_t1e1_set_dn(16, 0x4, "5551000");	this will set the default DN for B-channel 4 to 555-1000, on board 16

xds_t1e1_generate_ring

xds_t1e1_generate_ring(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value between 0 and the maximum number of B-channels on the board indicating which B-channel to control

Applicable Boards
T1 / E1 boards

Purpose
This function will generate ringing on a given B-channel.

Message Sent
“CRxx” where **xx** is the B-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The board will handle the ring cadence, and stop ringing if the far end goes off-hook. Ringing can be stopped by sending a message of the form **CDxx.** The ring cadence is the standard 2 seconds on, four seconds off.

Example

xds_t1e1_generate_ring(16, 2, 3);	this will generate ringing on port 2 of board 16.

xds_t1e1_enable_audio_channel

xds_t1e1_enable_audio_channel(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value between 0 and the maximum number of B-channels on the board indicating which B-channel to control

Applicable Boards
T1 / E1 boards

Purpose
This function will enable an audio channel on a given channel.

Message Sent
"CMxx" where **xx** is the B-channel

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an interface port was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will enable an audio channel with no change to the signaling bits.

Example

xds_t1e1_enable_audio_channel(16, 2, 3);	this will enable audio on channel 2 of board 16.

xds_t1e1_reset_bchannel

xds_t1e1_reset_bchannel(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	the number of the Layer 3 B-channel interface

Applicable boards
All XDS T1 / E1 boards

Purpose
This function is used to reset a B-channel at Layer 3. This involves clearing the call state and call reference for the selected B channel.

Message Sent
"RBxx" where **xx** is the B-channel number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function resets Layer 3 on the selected B-channel. The call state for that B-channel is set to idle (U0), and the associated call reference is released. This command does not disable connections to the CT bus.

Example
xds_t1e1_reset_bchannel(16, 0x4); this will reset B-channel 4 on board 16

xds_t1e1_system_options

xds_t1e1_system_options(board_number, option_a, option_b, option_c, option_d);

unsigned char board_number;	a value indicating which board the command is for
char option_a;	a value of ‘Y’ or ‘N’ indicating the selection
char option_b;	a value of ‘Y’ or ‘N’ indicating the selection
char option_c;	a value of ‘Y’ or ‘N’ indicating the selection
char option_d;	a value of ‘Y’ or ‘N’ indicating the selection

Applicable Boards
T1 / E1 boards

Purpose
This command is used to set optional Layer 3 behavior.

Message Sent
“SSabcdefgh” where **a** disables auto CONNECT ACK, **b** separates calling party # message, **c** controls the NSAP or user specified subaddress format, **d**

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_TYPE	an invalid system option was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Each option a-h can be set to either ‘Y’ or ‘N’ for yes or no. The default setting is ‘N’ for no.

The options are:
a - disable auto CONNECT ACK
b - separate calling party # message
c - NSAP or user specified subaddress format
d - disable Layer 3 timers
e-h reserved (for future use)

Example
xds_t1e1_system_options(16, ‘N’, ‘Y’, ‘N’, ‘Y’);

this will disable auto CONNECT ACK, use separate calling party # message, disable the NSAP or user specified subaddress format, and disable Layer 3 timers on board 16.

xds_t1e1_read_span_status

xds_t1e1_read_span_status(board_number, span, status);

unsigned char board_number;	a value indicating which board the command is for
int span;	the span to read from
struct xdst1e1 *status;	the data structure used to pass the status information

Applicable Boards
T1 / E1 boards

Purpose
This command is used to read Layer 1 and Layer 2 status information.

Message Sent
None

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Layer 1:
The alarm state of an interface can be determined in several ways. Each interface has a status LED associated with it, (See Sec. 3.3 of 257M016). This LED can be green, yellow, blue, red, or dark. Green indicates that there is no alarm condition detected. The other colors indicate the most serious alarm condition detected, i.e. red is more important than blue is more serious than yellow. Dark is reserved for undefined spans. The status of these LEDs is also present in dual-ported memory at an offset of 1D00h. There are 8 bytes, one for each span. Bits 0-3 represent the green, yellow, blue and red alarm states respectively. Note that bit 0 is always on for active interfaces. Bit 7 is set to a 1 on NT interfaces and to a 0 for TE and undefined interfaces.

Layer 2:
Each of the spans has eight bytes reserved for Layer 2 state information beginning at an offset of 1C00h. The first byte for a span represent the Layer 2 states for the data link, the other bytes are reserved. These states are:

1 - TEI unassigned
4 - TEI assigned, a TEI has been assigned, but multi-frame operation has not been established
5 - Awaiting multi-frame operation, an SABME frame has been sent and awaiting a UA frame acknowledgment
6 - Awaiting release from multi-frame state, a DISC frame has been sent and awaiting a DM frame
7 - Multi-frame operation, exchange of I frames is possible
8 - Timer recovery, a timer has expired and recovery procedures are in progress

Example
```
xds_t1e1_read_span_status(16, 0, status);
```

This will return a value in status->layer1_state and status->layer2_state indicating the span state information.

T1 Specific
Switching and Layer 3
Board Functions

This page was intentionally left blank.

xds_t1_play_call_progress

int xds_t1_play_call_progress(board_number, tone, stream, timeslot);

unsigned char board_number;	a value indicating which board the command is for
char tone;	a value indicating the call progress tone to be used
int stream;	a value indicating which stream is to be used
int timeslot;	a value indicating which timeslot is to be used

Applicable Boards
T1 PRI boards

Purpose
This function is used to give call progress tones on a given B-channel.

Message Sent
"CSxxsstt" where **xx** is the value of the tone to be used, **ss** is the stream, and **tt** is the timeslot

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

WRONG_BOARD	a non-T1 PRI board was selected
ILL_MODE	an invalid call progress tone was called for the board being used
ILL_SLOT	an invalid stream or timeslot was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The tone argument sent to the board is a hexadecimal value that corresponds to the parameter passed. An 0x18 would be sent if a tone parameter of '0' is passed, 0x19 would be sent if a '1' is passed, and so forth.
The valid modes for call progress are:
'0' – dial tone
'1' – reorder
'2' – busy signal
'3' – audible ringback
'4' – digital milliwatt
'5' – silence

Examples

xds_t1_play_call_progress(16, 11, '0');	give B-channel 11 a call progress "dial tone" tone on board 16
xds_t1_play_call_progress(16, 13, '2');	give B-channel 13 plays a call progress "busy signal" tone on board 16

xds_t1_l3_alert

xds_t1_l3_alert(board_number, b_channel, port_type, progress, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating the port type of the span used
char progress;	a character indicating the call progress indicator (NT span)
char *reference;	a pointer to a character string to be used as the call reference number (for CI spans)

Applicable boards
T1 PRI boards

Purpose
This function is used to send a Layer 3 ALERTing message for the call currently associated with the specified B-Channel. An ALERTing message is used to notify the caller that the called party is being informed of an incoming call.

Message Sent
NT span:
"DAxxp" where **xx** is the B-channel number and **p** is the progress indicator.

CI span:
"DAxxrrrr" where **xx** is the B-channel number and **rrrr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PORT_TYPE	a port type other than NT or CI has been chosen
ILL_PROGRESS	invalid progress indicator value
ILL_REFERENCE	out of range or NULL call reference number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an ALERTing message to be sent for the call associated with the specified B-channel.

A call progress indication can be sent as part of the message.

Valid values for the call progress indicator are:
C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

If the span is configured as CI, a call reference number can be sent as part of the message.

Valid values for the call reference number are between "0000" and "7FFF".

Example
NT:

xds_t1_l3_alert(16, 0, 'N', 'D', 0);	this will send an ALERTing message for B-channel 0 on board 16. The progress indicator is set for "Destination address is non-ISDN"

CI:

xds_t1_l3_alert(16, 2, 'C', 0, "0193");	this will send an ALERTing message for B-channel 2 on board 16. The call reference number is 0x193

xds_t1_l3_connect

xds_t1_l3_connect(board_number, b_channel, port_type, progress, reference, connected_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating the port type of the B-channel used
char progress;	a character indicating the progress indicator to use
char *reference;	a pointer to a character string to be used as the call reference number (for CI spans)
char *connected_number;	a pointer to a character string to be used as the connected number

Applicable boards
T1 PRI boards

Purpose
This function is used to send a Layer 3 CONNect message for the call currently associated with the specified B-Channel. A CONNect message is used to notify the caller that the called party has answered an incoming call.

Message Sent
NT span:
"DCxx(p)(#)" where **xx** is the B-channel number, **p** is the optional progress indicator, and the **#** is the optional connected number

CI span:
"DCxxp(rrrr)(#)" where **xx** is the B-channel number, **p** is the progress indicator, **rrrr** is the call reference number, and the **#** is the optional connected number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PORT_TYPE	a port type other than NT or CI has been chosen
ILL_REFERENCE	out of range call reference number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a CONNect message to be sent for the call associated with the specified B-channel. This should be done when the called party answers.

For any “optional” values, use a 0 (NULL value) to pass when not using them.

Valid values for the call reference number are between “0000” and “7FFF”.

Example
NT:

xds_t1_l3_connect(16, 0, ‘N’, “5551212”,);	this will send an CONNect message for B-channel 0 on board 16.

CI:

xds_t1_l3_connect(16, 2, ‘C’, “0193”);	this will send an CONNect message for B-channel 2 on board 16. The call reference number is 0x193

xds_t1_l3_disc_ref

xds_t1_l3_disc_ref(board_number, b_channel, cause, reference, connected_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating the port type of the B-channel used
int cause;	a value of the disconnect cause
char *reference;	a pointer to a character string to be used as the call reference number
char *connected_number;	a pointer to a character string to be used as the connected number

Applicable boards
T1 PRI boards

Purpose
This function is used to send a Layer 3 DISConnect message for the call currently associated with the specified B-Channel. A DISConnect message is used to notify either the network or the user that a disconnect sequence has been initiated.

Message Sent
"DDxxcc(rrrr)(#)" where **xx** is the B-channel number and **cc** is the cause code, **rrrr** is the optional call reference, and the **#** is the optional connected number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PORT_TYPE	a port type other than NT or CI has been chosen
ILL_CAUSE	invalid cause value
ILL_REFERENCE	invalid reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a DISConnect message to be sent for the call associated with the specified B-channel. This should be done when initiating a disconnect sequence.

The cause code is a value used to indicate the reason for a particular message, in this case, why a disconnect is occurring. A cause value of 0x10 indicates normal clearing. Other values may indicate error conditions. For a list of cause codes and their meanings, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

For any "optional" values, use a 0 (NULL value) to pass when not using them.

Valid values for the call reference number are between "0000" and "7FFF".

Example

xds_t1_l3_disconnect(16, 2, 0x10, "0102");	this will send a DISConnect message for B-channel 4 on board 16, with a cause of 0x10 (normal clearing) and call reference of 0x102.
xds_t1_l3_disconnect(16, 4, 0x10);	this will send a DISConnect message for B-channel 4 on board 16, with a cause of 0x10 (normal clearing)

xds_t1_l3_info_nt

xds_t1_l3_info_nt(board_number, b_channel, progress);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char progress;	a character indicating the state of the feature indicator

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to send a Layer 3 INFOrmation message from the network with optional cause, Keypad Facility, feature indication, blocking channel id or advise of charge elements.

Message Sent
"DIxxPp" where **xx** is the B-channel number and **p** is the call progress indicator

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFORMATION message to be sent from a port set to the network type.

Valid values for the call progress indicator are:
C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

Example

xds_t1_l3_info_nt(16, 4, ‘N’);	this will send an INFORMATION message for B-channel 4 on board 16 with a call progress value of ‘N’ for no progress indicator

xds_t1_l3_info_ci

xds_t1_l3_info_ci(board_number, b_channel, digits);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *digits;	a character string indicating the keypad digits

Applicable boards
T1 PRI boards (CI spans)

Purpose
This function is used to send a Layer 3 INFOrmation message from the network with optional cause, Keypad Facility, feature indication, blocking channel id or advise of charge elements.

Message Sent
"DIxxPp" where **xx** is the B-channel number and **p** is the call progress indicator

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid keypad digits selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFORMATION message to be sent from a port type set to CI.

Valid values for the digits are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, #, *

Example

xds_t1_l3_info_ci(16, 4, "1234");	this will send an INFORMATION message for B-channel 4 on board 16 with keypad digits "1234"

xds_t1_l3_notify

xds_t1_l3_notify(board_number, b_channel, notification, calling_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char notification;	a character indicating the notification type
char *calling_number;	a character string indicating the calling number

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to send a Layer 3 NOTIFY message to inform a user or terminal of the change of state of a call due to call rearrangement.

Message Sent
“DNxxn(#)” where **xx** is the B-channel number, **n** is the notification indicator, and **#** is the optional calling number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid notification indicator value or an invalid / NULL calling number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a NOTIFY message to be sent from an NT span to a user or terminal. The notification is used to inform the terminal of a change of state of a call due to call rearrangement so that the terminal may reflect the change of state by changing the state of an indicator (i.e. controlling whether an indicator blinks or not) or other means of informing the user.

The valid notification indicators are:

B -	Call bridged
C -	Call conferenced
D -	Privacy disabled
E -	Privacy enabled
F -	Call forwarded
H -	Call held
I -	Monitored user or resource idle
M -	Monitoring discontinued
R -	Call retrieved
T -	Call transfered
U -	Service profile update

Example

xds_t1_l3_notify(16, 4, ‘R’, 0);	this will send a NOTIFY message for B-channel 4 on board 16 with a notification of Call Retrieved.
xds_t1_l3_notify(16, 0, ‘B’, “5551212”);	this will send a NOTIFY message for B-channel 0 on board 16 with a notification of Call bridged, with a calling number of 555-1212.

xds_t1_l3_proceed

xds_t1_l3_proceed(board_number, b_channel, port_type, progress);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating the port type of the B-channel used
char progress;	a character indicating the progress indicator (NT span)

Applicable boards
T1 PRI Boards

Purpose
This function is used to send a Layer 3 CALL PROCeeding message for the call currently associated with the specified B-Channel. A CALL PROCeeding message is used to notify the caller that all information necessary to process a call has been received and that call establishment has been initiated.

Message Sent
NT span:
"DPxxp" where **xx** is the B-channel number and **p** is the progress indicator

CI span:
"DPxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PORT_TYPE	an invalid port type was selected for the B-channel
ILL_PROGRESS	invalid progress indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a CALL PROCeeding message to be sent for the call associated with the specified B-channel. This should be done in response to a SETUP message for Enbloc sending, or when all necessary digits have been received for Overlap sending when the call can be initiated.

Valid values for the call progress indicator are:
C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

Example

NT span:	
xds_t1_l3_proceed(16, 2, ‘N’, ‘R’);	this will send a CALL PROCeeding message for B-channel 2 on board 16, with a progress indicator of “Destination address is non-ISDN”
CI span:	
xds_t1_l3_proceed(16, 4, ‘C’, 0);	this will send a CALL PROCeeding message for B-channel 4 on board 16

xds_t1_l3_rel_com_ref

xds_t1_l3_rel_com_ref(board_number, b_channel, cause, reference, connected_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value of the disconnect cause
char *reference;	a pointer to a character string to be used as the call reference number
char *connected_number;	a character string indicating the connected number

Applicable boards
T1 PRI boards

Purpose
This function is used to send a Layer 3 REL_COMplete (with explicit reference) message for the call specified by the call reference. A REL_COMplete message is used to release a call reference and terminate a call in cases where a DISConnect message is inappropriate.

Message Sent
"DRxxcc(rrrr)(#)" where **xx** is the B-channel number, **cc** is the cause value, **rrrr** is the optional call reference, and **#** is the optional connected number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid connected number was used
ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	invalid call reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a REL_COMplete message to be sent for the call associated with the specified B-channel. This should be done to release a call and associated call reference under some circumstances. An example is when no TE equipment has responded to a SETUP message. If the call has been allocated a B-channel, a DISConnect should be sent to clear a call (see **xds_t1e1_l3_disconnect**).

The cause code is a value used to indicate the reason for the message. Examples would be a cause code of 0x10, "Normal clearing", or 0x41, "Bearer capability not supported". For a list of cause codes and their meaning, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

Valid values for the call reference number are between "0000" and "7FFF".

Example
xds_t1_l3_rel_com_ref(16, 4, 0x10, "0012", 0);

this will send a RELease COMplete message for B-channel 4 on board 16 with a cause value of 0x10 and a call reference of 0x12.

xds_t1_l3_rel_com_ref(16, 4, 0x10, "0012", "5551212");

this will send a RELease COMplete message for B-channel 4 on board 16 with a cause value of 0x10, a call reference of 0x12, and a connected number of 555-1212.

xds_t1_l3_setup_nt

xds_t1_l3_setup_nt(board_number, b_channel, bearer_cap, progress,
calling_number, called_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char bearer_cap;	a character indicating the type of call, i.e. speech or data
char progress;	a character indicating the progress indicator
char *calling_number;	a pointer to an ASCII string with the calling number
char *called_number;	a pointer to an ASCII string with the called number

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to send a Layer 3 SETUP message on an NT span. A SETUP message is used to initiate a call.

Message Sent
"DSxxbp#(/#)" where **xx** is the B-channel number, **b** is the bearer capability, **p** is the progress indicator, **#** is the calling number, and **/#** is the optional number being called.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
ILL_ARG	invalid bearer capability value, calling number, or called number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SETUP message to be sent to initiate a call. The treatment of the directory numbers is different for the two span types.

The bearer capability indicates the type of call being made, i.e. voice or data. There are four valid choices:
A - 3.1 kHz audio, 64 kbps, circuit mode, Mu-Law
D - Unrestricted digital information, 64 kbps, circuit mode
R - Rate adaption from 56 kbps, 64 kbps, circuit mode
S - Speech, 64 kbps, circuit mode, Mu-Law

Valid values for the call progress indicator are:

C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

The calling number element will use the default DN of the B-channel on which the call is being placed. The called number can be up to 15 digits long.

Examples
```
xds_t1_l3_setup(16, 2, ‘N’, ‘S’, ‘N’, “8384194”, “5551212”);
```

this will send a SETUP message for B-channel 2 on board 16 with a bearer capability of speech, no progress indicator, a calling number of 838-4194, and called number 555-1212.

xds_t1_l3_setup_ci

xds_t1_l3_setup_ci(board_number, b_channel, bearer_cap, progress,
calling_number, called_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char bearer_cap;	a character indicating the type of call, i.e. speech or data
char progress;	a character indicating the progress indicator
char *calling_number;	a pointer to an ASCII string with the calling number
char *called_number;	a pointer to an ASCII string with the called number

Applicable boards
T1 PRI boards (CI spans)

Purpose
This function is used to send a Layer 3 SETUP message on an NT span. A SETUP message is used to initiate a call.

Message Sent
"DSxxbp(#)/#" where **xx** is the B-channel number, **b** is the bearer capability, **p** is the progress indicator, **#** is the optional calling number, and **/#** is the number being called.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
ILL_ARG	invalid bearer capability value, calling number, or called number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SETUP message to be sent to initiate a call.

The bearer capability indicates the type of call being made, i.e. voice or data. There are four valid choices:
A - 3.1 kHz audio, 64 kbps, circuit mode, Mu-Law
D - Unrestricted digital information, 64 kbps, circuit mode
R - Rate adaption from 56 kbps, 64 kbps, circuit mode
S - Speech, 64 kbps, circuit mode, Mu-Law

Valid values for the call progress indicator are:

C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

The calling number element will use the default DN of the B-channel on which the call is being placed. The called number can be up to 15 digits long.

Examples

```
xds_t1_l3_setup_ci(16, 4, ‘T’, ‘D’, ‘’, “8384194” , “18005551212”);
```

this will send a SETUP message for B-channel 4 on board 16 with a bearer capability of data, calling number of 838-4194, and a called number of 1-800-555-1212.

xds_t1_l3_text

xds_t1_l3_text(board_number, mode, tag, text, submode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a value indicating the text operation command
char tag;	a character indicating the tag code used
char *text;	a pointer to a NULL terminated string of up to 128 characters
char submode;	a character indicating the submode used

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to prepare text to be displayed on the device used. Only NT (network termination) ports can send text.

Message Sent
"DTAt[text]" – Adds display text [**text**] to buffer with a display tag **t**
"DTC" – Clears display text buffer
"DTNt[text]" – Create new display text element [**text**] in buffer with a display tag **t**
"DTQA[text]" – Adds display (ANSI T1.607) text [**text**] to buffer
"DTQC[text]" – Clears display buffer
"DTQL[text]" – Creates a new display (ANSI T1.607) text [**text**] in buffer

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	invalid ANSI T1.607 mode
ILL_MODE	invalid text mode
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Multiple segments may be added as long as the total length of the display text element is less than 128 octets. Note, that as some octets are used for the segment tags and lengths that this represent fewer than 128 characters. Each segment includes a tag, which indicates what the nature of the text is, i.e. called party number, cause, or generic text. This has been assumed for the purposes of the "D" message set.

The "mode" controls which text operation is carried out. The valid mode values are:

'A'	add text to the buffer
'C'	clear the display text buffer
'N'	create a new display
'Q'	ANSI T1.607 display

The "submode" controls which text operation is carried out for the ANSI T1.607 modes. The valid submode values are:

'A'	add display text to buffer
'C'	clear the display buffer
'L'	create a new display

Examples

xds_t1_l3_text(16, 'C', '', '', 0);	this will clear the text buffer on board 16.
xds_t1_l3_text(16, 'A', '@', "Hello there", 0);	this will add the text "Hello there" to the buffer with the "text message" tag on board 16
xds_t1_l3_text(16, 'N', '{', "8384194", 0);	this will place "8384194" in the text buffer with the "Connected number" tag on board 16
xds_t1_l3_text(16, 'Q', '', '', 'C');	this will clear the text buffer on board 16.
xds_t1_l3_text(16, 'Q', '@', "Hello there", 'A');	this will add the text "Hello there" to the buffer with the "text message" tag on board 16

xds_t1_set_framing

int xds_t1_set_framing(board_number, span, framing, suppression, build_out, signalling);

unsigned char board_number;	a value indicating which board the command is for
int span;	a value indicating the span to control
char framing;	a character indicating the framing type to use
char suppression;	a character indicating suppression type to use
char build_out;	a character indicating the level of “build out” to use
char signalling;	a character indicating which signaling type to be used

Applicable Boards
T1 PRI boards

Purpose
This function is used to specify the framing format for each interface.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
ILL_TYPE	an invalid framing type was selected
ILL_MODE	an invalid suppression mode was selected
ILL_ARG	an out of range “build out” value was used
ILL_SIGNAL	an invalid signaling type was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Message Sent
“SFxxfzbs” where **xx** is the span (interface), **f** is the framing type, **z** is the suppression mode, **b** is the “build out” level, **s** is the signaling type.

Comments
This function is one of the functions to be used to set up the board before functional use.

This will set the framing parameters for each interface. The interface number is specified in xx.

The **f** parameter specifies the framing, ‘D’ for D4, or ‘E’ for ESF.

The **z** parameter specifies the zero suppression mode, ‘A’ for AMI, or ‘B’ for B8ZS.

The **b** parameter specifies the line build out.

Valid values are:
0 - DSX-1 (0-133ft.)/0dB CSU
1 - DSX-1 (133-266ft.)
2 - DSX-2 (266-399 ft.)
3 - DSX-3 (399 to 533 ft.)
4 - DSX-1 (533 to 655 ft.)
5 - -7.5 dB CSU
6 - -15 dB CSU
7 - -22 dB CSU

The **s** parameter specifies the signaling mode for the port, ‘R’ for robbed-bit signaling, ‘P’ for Primary Rate ISDN.

Example

xds_t1_set_framing(16, 0, ‘D’, ‘A’, 7, ‘R’);	sets framing on span 0 to D4, zero suppression mode AMI, -22 dB CSU for the build out, and robbed-bit signaling on board 16

xds_t1_l3_facility

xds_t1_l3_facility(board_number, b_channel, component, invoke_id, tag_value)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char component;	a value indicating the component value
int invoke_id;	a value to be used as the invoke ID code
int tag_value;	a value to be used as the invoke ID tag value

Applicable boards
T1 PRI boards

Purpose
This function is used to send a FACILITY message.

Message Sent
"DFxxcid(tt)" where **xx** is the B-channel number, **c** is the component value, **id** is the invoke ID code, and **tt** is the optional tag value

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid ID tag value
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY message to the board.

The valid range for the invoke ID is from 0 to 0x7F and ID tag value is from 1 to 0xFF.

The “component” types are:

‘I’	invoke
‘R’	return result
‘E’	return error
‘J’	reject

Example
xds_t1_l3_facility(16, 4, ‘E’, 0xFF, 0x19);

this will send a FACILITY message for B-channel 4, on board 16, with the return error component, an invoke ID of 0xFF, and a tag value of 0x19

xds_t1_l3_facility_path_replace

xds_t1_l3_facility_path_replace(board_number, b_channel, invoke_id,	**call_identity, redirecting_number)**
unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int invoke_id;	a value to be used as the invoke ID code
call_identity;	a value to be used as the call identity
redirecting_number;	a value to be used as the redirecting number

Applicable boards
T1 PRI boards

Purpose
This function is used to send a FACILITY path replace propose message.

Message Sent
"DFxxPid,c-#" where **xx** is the B-channel number, **id** is the invoke ID, **c** is the component value, **c** is the call identity number, and **-#** is the redirecting number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_MODE	invalid call identity value
ILL_ARG	invalid ID redirecting number
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command unction is used to send a FACILITY path replace propose message. The call identity (call_identity) can be more than one digit long.

Example
xds_t1_l3_facility_path_replace(16, 0, 4, "12", "6088384194");

this will send a FACILITY path replace propose message for B-channel 0, invoke ID of 4, on board 16, with call identity value of 12 and a redirecting number of 608-838-4194

xds_t1_l3_facility_notification_indicator

xds_t1_l3_facility_notification_indicator(board_number, b_channel, notification)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char notification;	a value to be used as the notification indicator

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to send a FACILITY name message.

Message Sent
"DfxxN(n)" where **xx** is the B-channel number, **id** is the invoke ID code, and **n** is the optional notification indicator.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid notification indicator value
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY name message to the board.

The valid notification indicators are:

B -	Call bridged	C -	Call conferenced
D -	Privacy disabled	E -	Privacy enabled
F -	Call forwarded	H -	Call held
I -	Monitored user or resource idle		
M -	Monitoring discontinued		
R -	Call retrieved	T -	Call transfered
U -	Service profile update		

Example
xds_t1_l3_facility_notification_indicator(16, 4, 'B');

this will send a FACILITY message for B-channel 4, on board 16, with an notification indicator value of Call bridged ('B')

xds_t1_l3_facility_rlt_ret_nt

xds_t1_l3_facility_rlt_ret_nt(board_number, b_channel, invoke_id, error)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int invoke_id;	a value to be used as the invoke ID code
int error;	a value to be used as the error code

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to send a FACILITY RLT return error message.

Message Sent
"DFxxLEidee" where **xx** is the B-channel number, **id** is the invoke ID code, and **ee** is the error code.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid error code
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY RLT return error message to the board.

The valid range for the invoke ID is from 0 to 0x7F.

The "error" codes are:
10 - RLT Bridge Fail
11 - RLT Call ID Not Found
12 - RLT Not Allowed
13 - RLT Switch Equipment Congestion

Example
xds_t1_l3_facility_rlt_ret_nt(16, 4, 0x7F, 0x12);

this will send a FACILITY RLT return error message for B-channel 4, on board 16, invoke ID of 0x7F, and an error code of 0x12 (RLT Not Allowed)

xds_t1_l3_facility_rlt_rej

xds_t1_l3_facility_rlt_rej(board_number, b_channel, invoke_id, problem)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int invoke_id;	a value to be used as the invoke ID code
int problem;	a value to be used as the problem code

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to send a FACILITY RLT reject message.

Message Sent
"DFxxLJidpp" where **xx** is the B-channel number, **id** is the invoke ID code, and **pp** is the problem code.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid problem code
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY RLT reject message to the board.

The valid range for the invoke ID is from 0 to 0x7F. The valid range for the problem code is from 0 to 0xFF.

Example
xds_t1_l3_facility_rlt_rej(16, 4, 0x0, 0x82);

this will send a FACILITY RLT reject message for B-channel 4, on board 16, invoke ID of 0x0, and a problem code of 0x82

xds_t1_l3_facility_rlt_3pa_nt

xds_t1_l3_facility_rlt_3pa_nt(board_number, b_channel)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to send a FACILITY Third Party accept message.

Message Sent
"DFxxLR02" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY Third Party accept message to the board.

Example
xds_t1_l3_facility_rlt_3pa_nt(16, 4);

this will send a FACILITY Third Party accept message for B-channel 4 on board 16

xds_t1_l3_facility_cd_ci

xds_t1_l3_facility_cd_ci(board_number, b_channel, invoke_id, reason, deflected_number)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int invoke_id;	a value to be used as the invoke ID code
char reason;	a value to be used as the deflection reason
char *deflected_number;	a value to be used as the deflected number

Applicable boards
T1 PRI boards (CI spans)

Purpose
This function is used to send a FACILITY (CD) message.

Message Sent
“DFxxDidr[#]” where **xx** is the B-channel number, **id** is the invoke ID code, **r** is the deflection reason, and **#** is the deflected number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid deflection reason
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY (CD) message to the board.

The valid range for the invoke ID is from 0 to 0x7F. The valid range for the problem code is from 0 to 0xFF.

The valid deflection reasons are:
‘B’ - user busy
‘D’ - call deflection
‘N’ - no response
‘U’ - unconditional
‘X’ - unknown

Example
xds_t1_l3_facility_cd_ci(16, 4, 0x7F, ‘U’, “5551234”);

this will send a FACILITY (CD) message for B-channel 4, on board 16, invoke ID of 0x7F, an unconditional deflection reason, and the deflected number is 555-1234

xds_t1_l3_facility_rlt_3pa_ci

xds_t1_l3_facility_rlt_3pa_ci(board_number, b_channel, callid)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *callid;	a value containing the call ID

Applicable boards
T1 PRI boards (CI spans)

Purpose
This function is used to send a FACILITY RLT transfer message.

Message Sent
"DFxxL<CallID>" where **xx** is the B-channel number and **CallID** is the call ID.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY RLT transfer message to the board.

Example
xds_t1_l3_facility_rlt_3pa_ci(16, 4, "091393");

this will send a FACILITY RLT transfer message for B-channel 4 on board 16 with a call ID of 091393

xds_t1_l3_facility_ssct

xds_t1_l3_facility_ssct(board_number, b_channel, invoke_id, mode, transfer_number, transferred_number)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int invoke_id;	a value to be used as the invoke ID code
char mode;	a value to be used as the receive mode
char *transfer_number;	a value to be used as the transfer number
char *transferred_number;	a value to be used as the transferred number

Applicable boards
T1 PRI boards (CI spans)

Purpose
This function is used to send a QSIG Single-Step Call Transfer (SSCT) FACILITY (CD) message.

Message Sent
"DFxxSidm#/#" where **xx** is the B-channel number, **id** is the invoke ID code, **r** is the deflection reason, and **#** is the deflected number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid transfer and/or transferred number(s) was used
ILL_MODE	invalid receive mode
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a QSIG single step call transfer FACILITY (CD) message to the board.

The valid range for the invoke ID is from 0 to 0x7F. The valid range for the problem code is from 0 to 0xFF.

The valid receive modes reasons are:
‘A’ - receive an ALERTing message
‘C’ - receive a CONNect message

Example
xds_t1_l3_facility_ssct(16, 4, 0x7F, ‘C’, “5551234”, “8885551235”);

this will send a FACILITY QSIG SSCT message for B-channel 4, on board 16, invoke ID of 0x7F, to receive a CONNect message, with a transfer number of 555-1234 and a transferred number of 888-555-1235

xds_t1_l3_facility_ect_ci

xds_t1_l3_facility_ect_ci(board_number, b_channel, invoke_id, transfer_mode, reference)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int invoke_id;	a value to be used as the invoke ID code
char transfer_mode;	a value that indicates whether the call is Enhanced ECT or Legacy ECT
char *reference;	a value to be used as the call reference number

Applicable boards
T1 PRI boards (CI spans)

Purpose
This function is used to send a FACILITY Explicit Call Transfer (ECT) message.

Message Sent
Enhanced ECT:
"DFxxTidrrrr" where **xx** is the B-channel number, **id** is the invoke ID code, and **rrrr** is the call reference number.

Legacy ECT:
"DFxxEidrrrr" where **xx** is the B-channel number, **id** is the invoke ID code, and **rrrr** is the call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_REFERENCE	out of range or NULL call reference number
ILL_MODE	invalid ECT call mode
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY Explicit Call Transfer message to the board.

The valid range for the invoke ID is from 0 to 0x7F. The valid range for the problem code is from 0 to 0xFF.

Valid values for the call reference number are between “0000” and “7FFF”.

Example
xds_t1_l3_facility_ect_ci(16, 4, 0x7F, ‘L’, “7FFF”);

this will send a FACILITY Legacy ECT message for B-channel 4, on board 16, invoke ID of 0x7F, and the call reference number is 0x7FFF

xds_t1_l3_facility_ect_ci(16, 4, 0x7F, ‘E’, “7FFF”);

this will send a FACILITY Enhanced ECT message for B-channel 4, on board 16, invoke ID of 0x7F, and the call reference number is 0x7FFF

xds_t1_l3_facility_ctactive

xds_t1_l3_facility_ctactive(board_number, b_channel, invoke_id, redirecting_number)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int invoke_id;	a value to be used as the invoke ID code
char *redirecting_number;	a value to be used as the redirecting number

Applicable boards
T1 PRI boards (NT and CI spans)

Purpose
This function is used to send a FACILITY (CT Active) message.

Message Sent
"DFxxAid[#]" where **xx** is the B-channel number, **id** is the invoke ID code, and **#** is the redirecting number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY (CT Active) message to the board.

The valid range for the invoke ID is from 0 to 0x7F. The valid range for the problem code is from 0 to 0xFF.

Example
xds_t1_l3_facility_ctactive(16, 4, 0x7F, "5551234");

this will send a FACILITY (CT Active) message for B-channel 4, on board 16, invoke ID of 0x7F, with a redirecting number of 555-1234

xds_t1_l3_facility_ctcomplete

xds_t1_l3_facility_ctcomplete(board_number, b_channel, invoke_id, redirection_number, alerting)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int invoke_id;	a value to be used as the invoke ID code
char *redirection_number;	a value to be used as the redirection number
char alerting;	a value to indicate if the call is in the alerting state

Applicable boards
T1 PRI boards (NT and CI spans)

Purpose
This function is used to send a FACILITY (CT Complete) message.

Message Sent
"DFxxCid[#](A)" where **xx** is the B-channel number, **id** is the invoke ID code, **#** is the redirection number, and 'A' if the call is in the alerting state or not.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid end designator value
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY (CT Complete) message to the board.

The valid range for the invoke ID is from 0 to 0x7F. The valid range for the problem code is from 0 to 0xFF.

Example
xds_t1_l3_facility_ctcomplete(16, 4, 0x7F, "5551234", 'N');
this will send a FACILITY (CT Complete) message for B-channel 4, on board 16, invoke ID of 0x7F, with a redirecting number of 555-1234, and the call is not in the alerting state.

xds_t1_l3_facility_ctcomplete(16, 4, 0x7F, "5551234", 'Y');
this will send a FACILITY (CT Complete) message for B-channel 4, on board 16, invoke ID of 0x7F, with a redirecting number of 555-1234, and the call is in the alerting state.

xds_t1_l3_facility_dli1

xds_t1_l3_facility_dli1(board_number, b_channel, invoke_id, reason, option, diverting_to_number)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int invoke_id;	a value to be used as the invoke ID code
char reason;	a value indicating the diversion reason
char option;	a value indicating the subscription option
char *redirecting_to_number;	the diverting to number

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to send a FACILITY Diverting Leg Information 1 message.

Message Sent
"DFxxF1idro#" where **xx** is the B-channel number, **id** is the invoke ID code, **r** is the reason, **o** is the subscription option, and **#** is the diverting to number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ID	invalid invoke ID code
ILL_MODE	invalid reason
ILL_TYPE	invalid subscription option
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY Diverting Leg Information 1 message to the board.

The valid range for the invoke ID is from 0 to 0x7F. The valid range for the problem code is from 0 to 0xFF.

The following values are valid for the diversion reason:
‘B’ - busy
‘D’ - deflected
‘N’ - no answer
‘U’ - unconditional

The following values are valid for the subscription option:
‘0’ – no notification
‘1’ – notification without diverted to number
‘2’ – notification with diverted to number

Example
xds_t1_l3_facility_dli1(16, 4, 0x7F, ’B’, ‘0’, “8384194”);

this will send a FACILITY diverting leg information 1 message for B-channel 4, on board 16, invoke ID of 0x7F, a reason of busy, no notification, and the diverting to number of 838-4194

xds_t1_l3_facility_dli1(16, 4, 0x7F, ’D’, ‘2’, “8384194”);

this will send a FACILITY diverting leg information 1 message for B-channel 4, on board 16, invoke ID of 0x7F, a reason of deflected, notification with diverted to number, and the diverting to number of 838-4194

xds_t1_l3_facility_dli3

xds_t1_l3_facility_dli3(board_number, b_channel, invoke_id)

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int invoke_id;	a value to be used as the invoke ID code

Applicable boards
T1 PRI boards (NT and CI spans)

Purpose
This function is used to send a FACILITY Diverting Leg Information 3 message.

Message Sent
"DFxxF3id" where **xx** is the B-channel number, and **id** is the invoke ID code.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a FACILITY Diverting Leg Information 3 message to the board.

The valid range for the invoke ID is from 0 to 0x7F. The valid range for the problem code is from 0 to 0xFF.

Example
xds_t1_l3_facility_dli3(16, 4, 0x7F, 'L', "7FFF");

this will send a FACILITY diverting leg information 3 message for B-channel 4, on board 16, invoke ID of 0x7F, and the call reference number is 0x7FFF

xds_t1_l3_include_rlt_op_ind

xds_t1e1_l3_include_rlt_op_ind(board_number);

unsigned char board_number;	a value indicating which board the command is for

Applicable boards
T1 PRI boards (CI spans)

Purpose
This function is used to put invoke RLT Operation Indication in buffer.

Message Sent
"D@R"

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used put invoke RLT Operation Indication in buffer.

Examples
xds_t1e1_l3_include_rlt_op_ind(16)

xds_t1_l3_include_rlt_call_id

xds_t1_l3_include_rlt_call_id(board_number, callid)

unsigned char board_number;	a value indicating which board the command is for
char *callid;	a value containing the call ID

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to send a put the RLT Call ID in the buffer.

Message Sent
"D@I<CallID>" where **CallID** is the call ID.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the length of the Call ID is not 6 or 8
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a put the RLT Call ID in the buffer.

Example
xds_t1_l3_include_rlt_call_id(16, "091393");

this will send a FACILITY RLT transfer message for B-channel 4 on board 16 with a call ID of 091393

xds_t1_l3_include_dli2

xds_t1_l3_include_dli2(board_number, invoke_id, diversion_count, reason, diversion_number)

unsigned char board_number;	a value indicating which board the command is for
int invoke_id;	a value to be used as the invoke ID code
char diversion_count;	a value indicating the diversion count
char reason;	a value to be used as the diversion reason
char *deflected_number;	a value to be used as the diversion number

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to send a put the diverting leg information 2 in the buffer.

Message Sent
"D@F2idcr#" where **id** is the invoke ID value, **c** is the diversion count, **r** is the reason for the diversion, and the **#** is the diversion number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the diversion number is invalid or too short
ILL_MODE	invalid diversion reason was used
ILL_ID	invalid invoke ID code
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a put the diverting leg information 2 in the buffer.

The following values are valid for the diversion reason:
'B' - busy
'D' - deflected
'N' - no answer
'U' - unconditional

Example
xds_t1_l3_include_dli2(16, 1, 3, 'B', "6088384194");

this will put a diverting leg information 2 message with an invoke ID of 01, diversion count of 3, reason of Busy, and a diversion number of 608-838-4194 in the buffer of board 16

xds_t1_l3_include_mwi_operation

xds_t1_l3_include_mwi_operation(board_number, invoke_id, operation, user_number, msg_center_number, message_waiting)

unsigned char board_number;	a value indicating which board the command is for
int invoke_id;	a value to be used as the invoke ID code
char operation;	a value indicating the operation mode
char *user_number;	a value to be used as the user number
char *msg_center_number;	a value to be used as the optional message center number
int message_waiting;	the optional number of messages waiting

Applicable boards
T1 PRI boards (CI spans)

Purpose
This function is used to send a put the MWI operation information in the buffer.

Message Sent
"D@Mido#(/#)(=m)" where **id** is the invoke ID value, **o** is the operation mode, the **#** is the user number, the **/#** is the optional message center number, and the **=m** is the number of messages waiting.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ID	invalid invoke ID code
ILL_MODE	the operation mode is invalid
ILL_ARG	the user number and/or the message center number is invalid
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to send a put the MWI operation information in the buffer. The message center number and the number of messages waiting are both optional arguments.

The following values are valid for the operation mode:
‘A’ - activate
‘D’ - deactivate

Example
```
xds_t1_l3_include_mwi_operation(16, 06, ‘A’, “8384194”, “5551212”, 4);
```

this will put an MWI operation information message with an invoke ID of 06, operation code ‘A’ to activate, user number of 838-4194, message center number of 555-1212, and a message waiting count of 4 in the buffer of board 16

xds_t1_l3_redirect_number

xds_t1_l3_redirect_number(board_number, reason, redirect_number)

unsigned char board_number;	a value indicating which board the command is for
char reason;	a value to be used as the redirect reason
char *redirect_number;	a value to be used as the redirect number

Applicable boards
T1 PRI boards (NT spans)

Purpose
This function is used to put redirecting number in buffer.

Message Sent
"D#Rr[#]" where **r** is the redirect reason, and **#** is the redirect number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid redirect reason was used
ILL_ARG	a NULL redirect number was passed to the function
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
To include a redirecting number in a SETUP message from an NT port, a command message of the form **D#Rr[#]** must be sent prior to the "DS" command for the SETUP message. The redirecting number and subaddress will be added to the SETUP message when it is sent.

The valid redirect reasons are:

A - all call redirected
B - number busy
D - call deflected
N - no answer
O - out of order
U - unknown

Example
xds_t1_redirect_number(16, 'B', "5551234");

this will put the redirecting number, 555-1234, into the buffer with a reason of "number busy"

xds_t1_net_specific_opts

xds_t1_net_specific_opts(board_number, span, carrier_id, facilities_code)

unsigned char board_number;	a value indicating which board the command is for
int span;	a value indicating the span to control
char *carrier_id;	a value to be used as the redirect number
int facilities_code;	a value to be used as the redirect reason

Applicable boards
T1 PRI boards

Purpose
This function is used to set up the Network Specific Facilities element for a given span.

Message Sent
"SAddabchh" where **dd** is the span, **abc** is the Carrier Identification Code, and **hh** is the facilities code.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the span selected is out of valid range
ILL_CODE	an invalid Carrier Identification Code value was used
ILL_ARG	a NULL Carrier Identification Code was passed to the function
ILL_VALUE	an invalid facilities code was passed to the function
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to set up the Network Specific Facilities element for a given span and can have a Carrier Identification Code ASCII value ranging from "000" to "999". The range for the facilities code can be from 0 to 0xFF.

Example
xds_t1_net_specific_opts(16, 1, "288", 0x93);

would set the network specific facilities to AT&T MEGACOMM for span 01 and have a facilities code of 0x93 s

This page was intentionally left blank.

E1 Specific / E1 ARINC Switching and Layer 3 Board Functions

This page was intentionally left blank.

xds_e1_set_framing

int xds_e1_set_framing(board_number, span, framing, suppression, signaling, impedance, v52_detection);

unsigned char board_number;	a value indicating which board the command is for
int span;	a value indicating the span to control
char framing;	a character indicating the framing type to use
char suppression;	a character indicating suppression type to use
char signaling;	a character indicating which signaling type to be used
char impedance;	a character indicating the amount of impedance to use
char v52_detection;	an optional character indicating whether or not to enable V5.2 detection

Applicable Boards
E1 PRA boards

Purpose
This function is used to specify the framing format for each interface.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
ILL_TYPE	an invalid framing type was selected
ILL_MODE	an invalid suppression mode was selected
ILL_ARG	an out of range "impedance" value was used
ILL_SIGNAL	an invalid signaling type was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Message Sent
"Sfxxfzsi(v)" where **xx** is the span (interface), **f** is the framing type, **z** is the suppression mode, **s** is the signaling type, **i** is the impedance level, and **v** is the optional V5.2 detection argument.

Comments
This function is one of the functions to be used to set up the board before functional use.

This will set the framing parameters for each interface. The interface number is specified in xx.

The **f** parameter specifies the framing, ‘C’ for CRC4, or ‘N’ for non-CRC4.

The **z** parameter specifies the zero suppression mode, ‘A’ for AMI, or ‘H’ for HDB3.

The **s** parameter specifies the signaling mode for the port. These are ‘N’ for no signaling, ‘C’ for Channel Associated Signaling, ‘P’ for Primary Rate Access ISDN signaling, and ‘S’ for use with Signaling System 7 when the signaling channel is carried on another span and timeslot 16 is made available for use as an audio path. Note that ‘N’ and ‘S’ are identical except for access to timeslot 16.

When the signaling mode is other than ‘S’, the range of channels available on a span runs from 00-1D with 00-0E being mapped to timeslots 1-15 on the span, and 0F-1D being mapped to timeslots 17-31. When the signaling mode is ‘S’, the range of channels runs from 00-1E, which are mapped linearly from timeslot 1-31, with channel 0F mapped to timeslot 16. This will affect all messages that use a channel as an argument.

The **i** parameter specifies the impedance level, ‘B’ for 75 ohms, or ‘R’ for 120 ohms.

Example

xds_e1_set_framing(16, 0, ‘C’, ‘A’, ‘C’, ‘R’, 0);	sets framing on span 0 to CRC4, zero suppression mode AMI, channel associated signaling, an impedance of 120 ohms, and no V5.2 detection enabled
xds_e1_set_framing(16, 0, ‘C’, ‘A’, ‘C’, ‘R’, ‘V’);	sets framing on span 0 to CRC4, zero suppression mode AMI, channel associated signaling, an impedance of 120 ohms, and V5.2 detection enabled

xds_e1_l3_alert

xds_e1_l3_alert(board_number, b_channel, port_type, progress, category, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating the port type of the span used
char progress;	a character indicating the call progress indicator
char category;	a character indicating the party category
char *reference;	a pointer to a character string to be used as the call reference number

Applicable boards
E1 PRA boards

Purpose
This function is used to send a Layer 3 ALERTing message for the call currently associated with the specified B-Channel. An ALERTing message is used to notify the caller that the called party is being informed of an incoming call.

Message Sent
NT span:
"Daxx(p)(pc)" where **xx** is the B-channel number, **p** is the optional progress indicator, and **pc** is the optional party category value.

TE span:
"DAxxp(rrrr)" where **xx** is the B-channel number and **rrrr** is the optional call reference number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the party category is not valid
ILL_PORT	the B-channel is out of valid range
ILL_PORT_TYPE	a port type other than NT or TE has been chosen
ILL_PROGRESS	invalid progress indicator value
ILL_REFERENCE	out of range or NULL call reference number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an ALERTing message to be sent for the call associated with the specified B-channel.

A call progress indication can be sent as part of the message.

Valid values for the call progress indicator are:
C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

Valid values for the party category are ‘E’, ‘X’, ‘O’, or ‘U’.

If the span is configured as NT, a call reference number can be sent as part of the message.

If the span is configured as TE, a party category can be sent as part of the message.

Valid values for the call reference number are between “0000” and “7FFF”.

Example
NT:

xds_e1_l3_alert(16, 0, ‘N’, ‘D’, ‘E’, 0);	this will send an ALERTing message for B-channel 0 on board 16. The progress indicator is set for “Destination address is non-ISDN”, with the party category set to ‘E’

TE:

xds_t1_l3_alert(16, 2, ‘T’, ‘O’, 0, “0193”);	this will send an ALERTing message for B-channel 2 on board 16. The progress indicator is set for “Origination address is non-ISDN” and the call reference number is 0x193

xds_e1_l3_connect

xds_e1_l3_connect(board_number, b_channel, port_type, progress, reference, category, connected_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating the port type of the B-channel used
char progress;	a character indicating the progress indicator to use
char *reference;	a pointer to a character string to be used as the call reference number (for CI spans)
char category;	a character indicating the party category
char *connected_number;	a pointer to a character string to be used as the connected number

Applicable boards
E1 PRA boards

Purpose
This function is used to send a Layer 3 CONNect message for the call currently associated with the specified B-Channel. A CONNect message is used to notify the caller that the called party has answered an incoming call.

Message Sent
NT span:
"DCxx(#cnct)(pc)" where **xx** is the B-channel number, **#cnct** is the optional connected number, and **pc** is the optional party category,

TE span:
"DCxxp(rrrr)(#cnct)" where **xx** is the B-channel number, **p** is the progress indicator, **rrrr** is the optional call reference number, and **#cnct** is the optional connected number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the party category is not valid or the connected number is invalid
ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
ILL_PORT_TYPE	a port type other than NT or TE has been chosen
ILL_REFERENCE	out of range call reference number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a CONNect message to be sent for the call associated with the specified B-channel. This should be done when the called party answers.

For any "optional" values, use a 0 (NULL value) to pass when not using them.

Valid values for the call progress indicator are:
C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

Valid values for the party category are 'E', 'X', 'O', or 'U'.

Valid values for the call reference number are between "0000" and "7FFF".

Example
NT:
```
xds_e1_l3_connect(16, 0, 'N', 'N', 0, "5550123" );
```

this will send an CONNect message for B-channel 0 on board 16, with no progress indicator, no party category, and a connected number of 555-0123

TE:
```
xds_e1_l3_connect(16, 2, 'T', 'W', "7FFF", 0, 0);
```

this will send an CONNect message for B-channel 2 on board 16 with a "delay in response at destination interface" progress indicator, call reference number of 0x7FFF, and no connected number

xds_e1_l3_info_keypad

xds_e1_l3_info_keypad(board_number, b_channel, digits);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *digits;	a character string indicating the keypad digits

Applicable boards
E1 PRA boards (TE spans)

Purpose
This function is used to send a Layer 3 INFOrmation message from the network with optional cause, Keypad Facility, feature indication, blocking channel id or advise of charge elements.

Message Sent
"DIxxPp" where **xx** is the B-channel number and **p** is the call progress indicator

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid keypad digits selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFORMATION message to be sent from a port type set to TE. This function calls xds_t1_l3_info_keypad(), from the T1 board chapter**.**

Valid values for the digits are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, #, *

Example

xds_e1_l3_info_keypad(16, 4, "1234");	this will send an INFORMATION message for B-channel 4 on board 16 with keypad digits "1234"

xds_e1_l3_disc_ref

xds_e1_l3_disc_ref(board_number, b_channel, cause, reference, connected_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating the port type of the B-channel used
int cause;	a value of the disconnect cause
char *reference;	a pointer to a character string to be used as the call reference number
char *connected_number;	a pointer to a character string to be used as the connected number

Applicable boards
T1 PRI boards

Purpose
This function is used to send a Layer 3 DISConnect message for the call currently associated with the specified B-Channel. A DISConnect message is used to notify either the network or the user that a disconnect sequence has been initiated.

Message Sent
NT span:
"DDxxcc(p)rrrr" where **xx** is the B-channel number and **cc** is the cause code, **p** is the optional call progress indicator, and **rrrr** is the call reference

TE span:
"DDxxcc(p)(rrrr)" where **xx** is the B-channel number and **cc** is the cause code, and **p** is the optional call progress indicator, and **rrrr** is the optional call reference

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_PROGRESS	invalid progress indicator value
ILL_REFERENCE	invalid reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a DISConnect message to be sent for the call associated with the specified B-channel. This should be done when initiating a disconnect sequence.

The cause code is a value used to indicate the reason for a particular message, in this case, why a disconnect is occurring. A cause value of 0x10 indicates normal clearing. Other values may indicate error conditions. For a list of cause codes and their meanings, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

For any “optional” values, use a 0 (NULL value) to pass when not using them.

Valid values for the call progress indicator are:
C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

Valid values for the call reference number are between “0000” and “7FFF”.

Example
NT span:
```
xds_e1_l3_disconnect(16, 2, 0x10, ‘O’, “0102”);
```

this will send a DISConnect message for B-channel 4 on board 16, with a cause of 0x10 (normal clearing), a progress indicator of origination address is non-ISDN and call reference of 0x102.

TE span:
```
xds_e1_l3_disconnect(16, 4, 0x10, ‘N’, 0);
```

this will send a DISConnect message for B-channel 4 on board 16, with a cause of 0x10 (normal clearing) – no call progress indicator, and no call reference number

xds_e1_l3_info

xds_e1_l3_info(board_number, b_channel, cause);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value of the disconnect cause

Applicable boards
E1 PRA boards

Purpose
This function is used to send a Layer 3 INFOrmation message from the network with cause

Message Sent
"DIxxcc" where **xx** is the B-channel number and **cc** is the cause

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFORMATION message to be sent from a port set to the network type.

Valid values for the cause are 0 – 0xFF:

Example

xds_e1_l3_info(16, 4, 'N');	this will send an INFORMATION message for B-channel 4 on board 16 with a call progress value of 'N' for no progress indicator

xds_e1_l3_info_keypad_facility

xds_e1_l3_info_keypad_facility(board_number, b_channel, digits);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *digits;	a string of digits

Applicable boards
E1 PRA boards (TE spans)

Purpose
This function is used to send a Layer 3 INFOrmation Keypad Facility digit message

Message Sent
"DIxxKk(k)" where **xx** is the B-channel number and **k(k)** are the digits

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid digits were selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFORMATION message with keypad facility digits

Valid values for the cause are 0 – 0xFF:

Example

xds_e1_l3_info_keypad_facilty(16, 4, "1234*#");	this will send an INFORMATION message for B-channel 4 on board 16 with digits 1234*#

xds_e1_l3_notify

xds_e1_l3_notify(board_number, b_channel, notification);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int notification;	a value indicating the notification indicator

Applicable boards
E1 PRA boards (NT spans)

Purpose
This function is used to send a Layer 3 NOTIFY message to inform a user or terminal of the change of state of a call due to call rearrangement.

Message Sent
"DNxxnn" where **xx** is the B-channel number, and **nn** is the notification indicator value

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	invalid notification indicator value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a NOTIFY message to be sent from an NT span to a user or terminal. The notification is used to inform the terminal of a change of state of a call due to call rearrangement so that the terminal may reflect the change of state by changing the state of an indicator (i.e. controlling whether an indicator blinks or not) or other means of informing the user.

The notification indicator value can be from 0 – 0xFF.

Example

xds_e1_l3_notify(16, 4, 0);	this will send a NOTIFY message for B-channel 4 on board 16 with a notification indicator value of 0

xds_e1_l3_notify_resume

xds_e1_l3_notify_resume(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
E1 PRA boards (NT spans)

Purpose
This function is used to send a Layer 3 NOTIFY call resumed message

Message Sent
“DNxxR” where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a NOTIFY call resumed message to be sent from an NT span to a user or terminal.

Example

xds_e1_l3_notify_resume(16, 4);	this will send a NOTIFY call resumed message for B-channel 4 on board 16

xds_e1_l3_notify_suspend

xds_e1_l3_notify_suspend(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
E1 PRA boards (NT spans)

Purpose
This function is used to send a Layer 3 NOTIFY call suspended message

Message Sent
“DNxxS” where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a NOTIFY call suspended message to be sent from an NT span to a user or terminal.

Example

xds_e1_l3_notify_suspend(16, 4);	this will send a NOTIFY call suspended message for B-channel 4 on board 16

xds_e1_l3_proceed

xds_e1_l3_proceed(board_number, b_channel, port_type, progress, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char port_type;	a character indicating the port type of the B-channel used
char progress;	a character indicating the progress indicator
char *reference;	a pointer to a character string to be used as the call reference number

Applicable boards
E1 PRA Boards

Purpose
This function is used to send a Layer 3 CALL PROCeeding message for the call currently associated with the specified B-Channel. A CALL PROCeeding message is used to notify the caller that all information necessary to process a call has been received and that call establishment has been initiated.

Message Sent
NT span:
"DPxx(p)" where **xx** is the B-channel number and **p** is the progress indicator

TE span:
"DPxx(p)(rrrr)" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
ILL_REFERENCE	out of range or NULL call reference number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a CALL PROCeeding message to be sent for the call associated with the specified B-channel. This should be done in response to a SETUP message for Enbloc sending, or when all necessary digits have been received for Overlap sending when the call can be initiated.

Valid values for the call progress indicator are:
C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

Valid values for the call reference number are between “0000” and “7FFF”.

Example

NT span:	
xds_e1_l3_proceed(16, 2, ‘N’, ‘R’, 0);	this will send a CALL PROCeeding message for B-channel 2 on board 16, with a progress indicator of “Destination address is non-ISDN”
TE span:	
xds_e1_l3_proceed(16, 4, ‘T’, ‘N’, “1993”)	this will send a CALL PROCeeding message for B-channel 4 on board 16 with a call reference number of 0x1993 and no progress indicator

xds_e1_l3_info_complete

xds_e1_l3_info_complete(board_number, b_channel, digits);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char *digits;	a string of digits

Applicable boards
E1 PRA boards (TE spans)

Purpose
This function is used to send a Layer 3 INFOrmation Keypad Facility digit message

Message Sent
"DKxxC(k)" where **xx** is the B-channel number and **k** is the digits

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_ARG	invalid digits were selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause an INFORMATION complete message with keypad digits

Valid values for the cause are 0 – 0xFF:

Example
xds_e1_l3_info_complete(16, 4, "1234*#");

this will send an INFORMATION message for B-channel 4 on board 16 with digits 1234*#

xds_e1_l3_query_sub_number

xds_e1_l3_query_sub_number (board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
E1 PRA boards (NT spans)

Purpose
This function is used to query the default subscriber number of a B-channel

Message Sent
“DQxx” where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to query the default subscriber number of a B-channel

Example
xds_e1_l3_query_sub_number(16, 9);

This will query the default subscriber number of B-channel 9

xds_e1_l3_rel_com_ref

xds_e1_l3_rel_com_ref(board_number, b_channel, cause, reference);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
int cause;	a value of the disconnect cause
char *reference;	a pointer to a character string to be used as the call reference number

Applicable boards
E1 PRA boards

Purpose
This function is used to send a Layer 3 REL_COMplete (with explicit reference) message for the call specified by the call reference. A REL_COMplete message is used to release a call reference and terminate a call in cases where a DISConnect message is inappropriate.

Message Sent
"DRxxcc(rrrr)" where **xx** is the B-channel number, **cc** is the cause value, and **rrrr** is the optional call reference

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_CAUSE	invalid cause value
ILL_REFERENCE	invalid call reference value
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a REL_COMplete message to be sent for the call associated with the specified B-channel. This should be done to release a call and associated call reference under some circumstances. An example is when no TE equipment has responded to a SETUP message. If the call has been allocated a B-channel, a DISConnect should be sent to clear a call (see **xds_t1e1_l3_disconnect**).

The cause code is a value used to indicate the reason for the message. Examples would be a cause code of 0x10, “Normal clearing”, or 0x41, “Bearer capability not supported”. For a list of cause codes and their meaning, see the XDS Layer 3 Protocol Software Reference Manual or Q.850.

Valid values for the call reference number are between “0000” and “7FFF”.

Example
xds_e1_l3_rel_com_ref(16, 4, 0x10, 0);

this will send a RELease COMplete message for B-channel 4 on board 16 with a cause value of 0x10 and no call reference

xds_e1_l3_rel_com_ref(16, 4, 0x10, “0012”);

this will send a RELease COMplete message for B-channel 4 on board 16 with a cause value of 0x10, and a call reference of 0x12

xds_e1_l3_setup_complete

xds_e1_l3_setup_complete(board_number, b_channel, bearer_cap, progress, calling_number, called_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char bearer_cap;	a character indicating the type of call, i.e. speech or data
char progress;	a character indicating the progress indicator
char *calling_number;	a pointer to an ASCII string with the calling number
char *called_number;	a pointer to an ASCII string with the called number

Applicable boards
E1 PRA boards

Purpose
This function is used to send a Layer 3 SETUP, sending complete, message. A SETUP message is used to initiate a call.

Message Sent
"DSxxbp(#)C#" where **xx** is the B-channel number, **b** is the bearer capability, **p** is the progress indicator, **#** is the optional calling number, and the second **#** is the number being called.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
ILL_ARG	invalid bearer capability value, calling number, or called number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SETUP message to be sent to initiate a call.

The bearer capability indicates the type of call being made, i.e. voice or data. There are four valid choices:
A - 3.1 kHz audio, 64 kbps, circuit mode, Mu-Law
D - Unrestricted digital information, 64 kbps, circuit mode
R - Rate adaption from 56 kbps, 64 kbps, circuit mode
S - Speech, 64 kbps, circuit mode, Mu-Law

Valid values for the call progress indicator are:

C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

The calling number element will use the default DN of the B-channel on which the call is being placed. The called number can be up to 15 digits long.

Examples
xds_e1_l3_setup_complete(16, 4, ‘T’, ‘D’, ‘’, “8384194” , “18005551212”);

this will send a SETUP message for B-channel 4 on board 16 with a bearer capability of data, calling number of 838-4194, and a called number of 1-800-555-1212.

xds_e1_l3_setup_te

xds_e1_l3_setup_te(board_number, b_channel, bearer_cap, progress,
calling_number, called_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char bearer_cap;	a character indicating the type of call, i.e. speech or data
char progress;	a character indicating the progress indicator
char *calling_number;	a pointer to an ASCII string with the calling number
char *called_number;	a pointer to an ASCII string with the called number

Applicable boards
E1 PRA boards (TE spans)

Purpose
This function is used to send a Layer 3 SETUP message on a TE span. A SETUP message is used to initiate a call.

Message Sent
"DSxxbp(#)(/#)" where **xx** is the B-channel number, **b** is the bearer capability, **p** is the progress indicator, **#** is the optional calling number, and the second **#** is the number being called.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
ILL_ARG	invalid bearer capability value, calling number, or called number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SETUP message to be sent to initiate a call.

The bearer capability indicates the type of call being made, i.e. voice or data. There are four valid choices:
A - 3.1 kHz audio, 64 kbps, circuit mode, Mu-Law
D - Unrestricted digital information, 64 kbps, circuit mode
R - Rate adaption from 56 kbps, 64 kbps, circuit mode
S - Speech, 64 kbps, circuit mode, Mu-Law

Valid values for the call progress indicator are:

C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

The calling number element will use the default DN of the B-channel on which the call is being placed. The called number can be up to 15 digits long.

Examples
xds_t1_l3_setup_te(16, 4, ‘T’, ‘D’, ‘’, “8384194” , “18005551212”);

this will send a SETUP message for B-channel 4 on board 16 with a bearer capability of data, calling number of 838-4194, and a called number of 1-800-555-1212.

xds_e1_l3_setup_nt

xds_e1_l3_setup_nt(board_number, b_channel, bearer_cap, progress, calling_number, called_number);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char bearer_cap;	a character indicating the type of call, i.e. speech or data
char progress;	a character indicating the progress indicator
char *calling_number;	a pointer to an ASCII string with the calling number
char *called_number;	a pointer to an ASCII string with the called number

Applicable boards
E1 PRA boards

Purpose
This function is used to send a Layer 3 SETUP message on an NT span. A SETUP message is used to initiate a call.

Message Sent
"DSxxbp(#)/#" where **xx** is the B-channel number, **b** is the bearer capability, **p** is the progress indicator, **#** is the optional calling number, and **/#** is the number being called.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	the B-channel is out of valid range
ILL_PROGRESS	invalid progress indicator value
ILL_ARG	invalid bearer capability value, calling number, or called number
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This command is used to cause a SETUP message to be sent to initiate a call. This function calls the **xds_t1_l3_setup_ci()** function from the T1 chapter.

The bearer capability indicates the type of call being made, i.e. voice or data. There are four valid choices:
A - 3.1 kHz audio, 64 kbps, circuit mode, Mu-Law
D - Unrestricted digital information, 64 kbps, circuit mode
R - Rate adaption from 56 kbps, 64 kbps, circuit mode
S - Speech, 64 kbps, circuit mode, Mu-Law

Valid values for the call progress indicator are:
C - Call is not end-to-end ISDN, call progress information may be available inband.
D - Destination address is non-ISDN
O - Origination address is non-ISDN
I - Inband information or appropriate pattern is now available
W - Delay in response at destination interface
N - no progress indicator

The calling number element will use the default DN of the B-channel on which the call is being placed. The called number can be up to 15 digits long.

Examples
xds_e1_l3_setup_nt(16, 4, ‘T’, ‘D’, ‘’, “8384194” , “18005551212”);

this will send a SETUP message for B-channel 4 on board 16 with a bearer capability of data, calling number of 838-4194, and a called number of 1-800-555-1212.

xds_e1_l3_text

xds_e1_l3_text(board_number, mode, tag, text);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a value indicating the text operation command
char tag;	a character indicating the tag code used
char *text;	a pointer to a NULL terminated string of up to 128 characters

Applicable boards
E1 PRA boards

Purpose
This function is used to prepare text to be displayed on the device used. Only NT (network termination) ports can send text.

Message Sent
"DTxxA(text)" – Adds display text (**text)** to buffer
"DTxxC" – Clears display text buffer
"DTxxL(text)" – Create new display text element (**text)** in buffer

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	the text string was NULL
ILL_MODE	invalid text mode
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Multiple segments may be added as long as the total length of the display text element is less than 128 octets. Note, that as some octets are used for the segment tags and lengths that this represent fewer than 128 characters. Each segment includes a tag, which indicates what the nature of the text is, i.e. called party number, cause, or generic text. This has been assumed for the purposes of the "D" message set.

The "mode" controls which text operation is carried out. The valid mode values are:

'A'	add text to the buffer
'C'	clear the display text buffer
'N'	create a new display
'Q'	ANSI T1.607 display

Examples

xds_e1_l3_text(16, 0, 'C', 0);	this will clear the text buffer on board 16, B-channel 0
xds_e1_l3_text(16, 0x0F, 'A', "Hello there");	this will add the text "Hello there" to the buffer on board 16, B-channel 15
xds_e1_l3_text(16, 0x01, 'N', "8384194");	this will place "8384194" in the text buffer on board 16, B-channel 1

xds_e1_mfcr2_generate_forward_tone

xds_e1_mfcr2_generate_forward_tone(board_number, b_channel, tone);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char tone;	a value indicating the character representing the frequency of the MF-R2 forward tone to be used

Applicable boards
E1 PRA boards

Purpose
This function is used to generate an MF-R2 forward tone on a given B-channel on an XDS E1 interface board.

Message Sent
“FFxxt” where **xx** is the B-channel number and the **t** is the tone character representing the frequency to be used as the forward tone to be generated

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
ILL_MODE	invalid backward tone value was selected by user
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Please refer to **Chapter 19, Addendum 'A'** (at the end of this chapter) for additional comments about this function and all other MF-R2 library functions. Below listed is a table for the tones available for this function. '0' – 'E' are the tones associated with the signals (I1 – II15 and A1 – B15) used. The user will pass a '0' – 'E' to the function.

Tone	**I**	**II**	**A**	**B**
1	I1	II1	A1	B1
2	I2	II2	A2	B2
3	I3	II3	A3	B3
4	I4	II4	A4	B4
5	I5	II5	A5	B5
6	I6	II6	A6	B6
7	I7	II7	A7	B7
8	I8	II8	A8	B8
9	I9	II9	A9	B9
0	I10	II10	A10	B10
A	I11	II11	A11	B11
B	I12	II12	A12	B12
C	I13	II13	A13	B13
D	I14	II14	A14	B14
E	I15	II15	A15	B15

Examples
```
xds_e1_mfcr2_generate_forward_tone(16, 0, '1');
```
this will generate an I1 or II1 (depends on specification use) tone on B-channel 0, on board 16

```
xds_e1_mfcr2_generate_forward_tone(16, 0, '0');
```
this will generate an I10 or II10 (depends on specification use) tone on B-channel 0, on board 16

xds_e1_mfcr2_generate_backward_tone

xds_e1_mfcr2_generate_backward_tone(board_number, b_channel, tone);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control
char tone;	a value indicating the character representing the frequency of the MF-R2 backward tone to be used

Applicable boards
E1 PRA boards

Purpose
This function is used to generate an MF-R2 backward tone on a given B-channel on an XDS E1 interface board.

Message Sent
"FBxxt" where **xx** is the B-channel number and the **t** is the tone character representing the frequency to be used as the backward tone to be generated

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
ILL_MODE	invalid backward tone value was selected by user
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Please refer to **Chapter 19, Addendum 'A'** (at the end of this chapter) for additional comments about this function and all other MF-R2 library functions. Below listed is a table for the tones available for this function. '0' – 'E' are the tones associated with the signals (I1 – II15 and A1 – B15) used. The user will pass a '0' – 'E' to the function.

Tone	**I**	**II**	**A**	**B**
1	I1	II1	A1	B1
2	I2	II2	A2	B2
3	I3	II3	A3	B3
4	I4	II4	A4	B4
5	I5	II5	A5	B5
6	I6	II6	A6	B6
7	I7	II7	A7	B7
8	I8	II8	A8	B8
9	I9	II9	A9	B9
0	I10	II10	A10	B10
A	I11	II11	A11	B11
B	I12	II12	A12	B12
C	I13	II13	A13	B13
D	I14	II14	A14	B14
E	I15	II15	A15	B15

Examples
```
xds_e1_mfcr2_generate_backward_tone(16, 0, '1');
```
this will generate an A1 or B1 (depends on specification use) tone on B-channel 0, on board 16

```
xds_e1_mfcr2_generate_backward_tone(16, 0, '0');
```
this will generate an A10 or B10 (depends on specification use) tone on B-channel 0, on board 16

xds_e1_mfcr2_enable_forward_tone

xds_e1_mfcr2_enable_forward_tone(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
E1 PRA boards

Purpose
This function is used to enable forward MF detection on a given B-channel on an XDS E1 interface board.

Message Sent
“FLxx” where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Please refer to **Chapter 19, Addendum ‘A’** (at the end of this chapter) for additional comments about this function and all other MF-R2 library functions.

Examples
xds_e1_mfcr2_enable_forward_tone(16, 0);
this will enable forward MF detection on B-channel 0, on board 16

xds_e1_mfcr2_disable_forward_tone

xds_e1_mfcr2_enable_backward_tone(board_number, b_channel);

unsigned char board_number;	a value indicating which board the command is for
int b_channel;	a value indicating the B-channel to control

Applicable boards
E1 PRA boards

Purpose
This function is used to disable forward MF detection on a given B-channel on an XDS E1 interface board.

Message Sent
"FDxx" where **xx** is the B-channel number

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid B-channel value was called for the board being used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Please refer to **Chapter 19, Addendum 'A'** (at the end of this chapter) for additional comments about this function and all other MF-R2 library functions.

Examples
xds_e1_mfcr2_disable_forward_tone(16, 0);
this will disable forward MF detection on B-channel 0, on board 16

Chapter 19, Addendum A

The MF signal consists of pairs of six different frequencies. There are fifteen possible tone pairs. A different set of frequencies is used for the forward and backward direction. The forward direction is from the end that is originating the call, while the backward signals are sent from the end receiving the call. The forward signals are referred to as I1 through I15 and II1 through II15 while there backward signals are A1 through A15 and B1 through B15. I1 and II1 are the same tone pair, but have different meanings depending on the call context. Similarly A1 and B1 are the same tones, but have different meanings. Some of the meanings are defined by national standards. In the commands and responses, the characters 0-9 and A-E represent the varios signals as given in the following table

Character	**I**	**II**	**A**	**B**
1	I1	II1	A1	B1
2	I2	II2	A2	B2
3	I3	II3	A3	B3
4	I4	II4	A4	B4
5	I5	II5	A5	B5
6	I6	II6	A6	B6
7	I7	II7	A7	B7
8	I8	II8	A8	B8
9	I9	II9	A9	B9
0	I10	II10	A10	B10
A	I11	II11	A11	B11
B	I12	II12	A12	B12
C	I13	II13	A13	B13
D	I14	II14	A14	B14
E	I15	II15	A15	B15

The signals I1-I9 represent the digits 1-9 and I10 represents the digit 0. The signal A1 means "send next digit" while A3 means "address complete"

A forward signal must be acknowledged. It will play until such time as a backward signal is detected. It will then stop which will cause the backward signal to stop. The next forward signal can then proceed. In certain cases the signals A3, A4, A6 or A15 may be sent without a forward signal being present.

To detect digits, the detector must be enabled with an **FLxx** command. Detection will continue until an **FDxx** command is issued. When a forward signal is recieved, an **FFxxt** response will be sent. A backward signal is sent with an **FBxxt** command.

As an example, the following messages would indicate the number 555-1234. The example is from the receiving side.

rcv:	**SF00**	off hook on 00
xmt:	**FL00**	enable MF detection
rcv:	**FF005**	receive digit 5
xmt:	**FB001**	acknowledge with an A1
rcv:	**FF005**	receive digit 5
xmt:	**FB001**	acknowledge with an A1
rcv:	**FF005**	receive digit 5
xmt:	**FB001**	acknowledge with an A1
rcv:	**FF001**	receive digit 1
xmt:	**FB001**	acknowledge with an A1
rcv:	**FF002**	receive digit 2
xmt:	**FB001**	acknowledge with an A1
rcv:	**FF003**	receive digit 3
xmt:	**FB001**	acknowledge with an A1
rcv:	**FF004**	receive digit 4
xmt:	**FB003**	acknowledge with an A3
xmt:	**FD00**	disable MF detection

When originating a call, forward tones are sent with an **FFxxt** command. The backward tones are indicated with an **FBxxt** response message. When the FB message is received, the next forward signal can be sent. It is not necessary to enable a detector for forward signaling as this is done automatically. If the forward signal is not acknowledged, it will stop after 4 sec. An **SExx** response will be sent indicating that the tone stopped because of a timeout.

Line signaling, that is on and off hook, is normally done with the A, B, C, and D bits using CAS, channel associated signaling. To do this, the span must be set up for CAS. Typically, the B-channel protocol should be set to Q for Q.421.

An example of a complete call is in the Infinity Series H.100 E1/Primary Rate ISDN Board Technical Manual, Amtelco P/N 257M016. Also, for more information regarding to the MFCR2 specification, please refer to the Q.400 series specification documentation.

SCSA Multi-Chassis (SCx) Board Functions

This page was intentionally left blank.

xds_query_sc

xds_query_sc(board_number, sc_rcv_timeslot, &query);

unsigned char board_number;	a value indicating which board the command is for
int sc_rcv_timeslot;	a value between 0-1023 indicating which timeslot on the SCbus is serving as an input to sample
struct rcvmsg *query{	
unsigned char board_number;	a value indicating which board the command is for
char msg_type;	a value denoting the message type, 16
char msg_sub;	a value denoting the message subtype, 9
int port_number;	the port number, (-1)
int timeslot;	the SCbus timeslot number
unsigned char argument;	returns with the sample value
char msg_str[32];	a character array, NULL
}	

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function is used to read the signal on an SCbus timeslot.

Message Sent
"QIxxx" where **xxx** is the SCbus timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was used
WRONG_BOARD	a different board has received the message
WRONG_QUERY	a different query response message was returned
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

The timeslot sample data is returned in the argument variable of the struct query if the function is successful.

Comments
The sample value will have a range of 0x0 to 0xFF. It is obtained by making a request of the board. The board will get the data and return it in a message which is then read and the data placed in the struct query. There may be a delay of as long as a hundred milliseconds between the function call and the return of the data. This delay is managed by the driver. The message from the board is placed on the query queue. If any messages are on that queue before **xds_query_sc** is called, an error may result. Because of this delay, this function can not be used to continuously read changing data from an SCbus timeslot.

Example

xds_query_sc(1, 2, &query);	this queries board 1 for the value of the data in timeslot 2 of the SCbus

the function will return with the following data in the struct query:

query.board_number = 1
query.msg_type = 16
query.msg_sub = 9
query.port_number = -1
query.timeslot = 2
query.argument = sample data
query.msg_msg = response message

xds_query_scx

xds_query_scx(board_number, scx_rcv_timeslot, &query);

unsigned char board_number;	a value indicating which board the command is for
int scx_rcv_timeslot;	a value between 0-1535 indicating which timeslot on theSCxbus is serving as an input to sample
struct rcvmsg *query{	
unsigned char board_number;	a value indicating which board the command is for
char msg_type;	a value denoting the message type, 16
char msg_sub;	a value denoting the message subtype, 9
int port_number;	the port number, (-1)
int timeslot;	the SCxbus timeslot number
unsigned char argument;	returns with the sample value
char msg_str[32];	a character array, NULL
}	

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function is used to read the signal on an SCxbus timeslot.

Message Sent
"QOzzzz" where **zzzz** is the SCxbus timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was used
WRONG_BOARD	a different board has received the message
WRONG_QUERY	a different query response message was returned
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

The timeslot sample data is returned in the argument variable of the struct query if the function is successful.

Comments
The sample value will have a range of 0x0 to 0xFF. It is obtained by making a request of the board. The board will get the data and return it in a message which is then read and the data placed in the struct query. There may be a delay of as long as a hundred milliseconds between the function call and the return of the data. This delay is managed by the driver. The message from the board is placed on the query queue. If any messages are on that queue before **xds_query_scx** is called, an error may result. Because of this delay, this function can not be used to continuously read changing data from an SCxbus timeslot.

Example

xds_query_scx(1, 1535, &query);	this queries board 1 for the value of the data in timeslot 1535 of the SCxbus

the function will return with the following data in the struct query:

```
query.board_number = 1
query.msg_type = 16
query.msg_sub = 9
query.port_number = -1
query.timeslot = 1535
query.argument = sample data
query.msg_msg = response message
```

xds_sample_position

xds_sample_position(board_number, position);

unsigned char board_number;	a value indicating which board the command is for
char position;	a character indicating whether the normal or middle sample position is used, ‘N’ selects normal and ‘M’ selects middle

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function is used to select the input sample position. The default sample position is at the end of a bit. However, under some circumstances it may be necessary to sample an input at the 3/4 position.

Message Sent
“SSN” for normal sample position, “SSM” for the mid or 3/4 position.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The default sample position is at the end of a bit. However, if one or more of the boards connected to the SCxbus is an XDS MVIP Multi-Chassis Board (MC1), it is necessary to change the sample position to the mid or 3/4 bit position.

Example

xds_sample_position(1, ‘M’);	this selects the 3/4 sample position.
xds_sample_position(1, ‘N’);	this selects the normal or end sample position.

xds_sc_pattern

xds_sc_pattern(board_number, port, pattern);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port is outputting to the SCbus. Ports on the Multi-chassis Board are actually virtual entities related to switching between the SCx and SC buses.
unsigned char pattern;	a value between 0x0 and 0xFF to be output to the SCbus

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function may be used to output a fixed pattern on the SCbus. This may be used for diagnostic purposes or to place "silence" or some other pattern on a particular timeslot.

Message Sent
"XPIxxxpp" where **xxx** is the port number and pp is the pattern value.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an port number was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function will cause the pattern value to be output to the SCbus timeslot which is defined by the base timeslot set for the board at initialization and the port number. This timeslot may be determined using the **xds_xmt_timeslot** function. Placing a pattern on a timeslot may be used as a diagnostic tool to check bus integrity. It may also be used to place a "silence" pattern of 0xFF on the SCbus. A pattern may be disabled by using the **xds_sc_rls** function.

Example

xds_sc_pattern(1, 2, 0xFF);	this outputs the pattern 0xFF on the timeslot reserved for port 2 on board 1.

xds_sc_rls

xds_sc_rls(board_number, port);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port is outputting to the SCbus. Ports on the Multi-chassis Board are actually virtual entities related to switching between the SCx and SC buses.

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function will release a connection between the SCxbus and the SCbus and disable the output on the SCbus timeslot reserved for the port to transmit on. This function will break down only half of a full duplex connection.

Message Sent
“XDIxxx” where **xxx** is the SCbus port number.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_PORT	an invalid port was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The **xds_sc_rls** function releases a connection from the SCxbus to the SCbus that has been established using either the **xds_scx_connect** or **xds_scx_listen** functions. To release a connection from the SCbus to the SCxbus, the **xds_scx_rls** function must be used. Therefore, to release a full duplex connection both the **xds_sc_rls** and **xds_scx_rls** functions must be used.

Example

xds_sc_rls(1, 2);	this will disable the SCbus output from port 2 on board 1.

xds_sc_select

xds_sc_select(board_number, streams);

unsigned char board_number;	a value indicating which board the command is for
char *streams;	a pointer to a character array indicating which SCbus streams to enable

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function is used to select which SCbus streams the Multi-Chassis board can transmit on. Hardware limitations restrict the board to transmitting on, at most, eight streams. This function is only needed when running non 4.1SC compatible programs.

Message Sent
"SBabcdefgh" where **abcdefgh** is the stream selection information.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
Normally, this function is not necessary as SCbus streams are enabled during the initialization and timeslot assignment phase at boot time. However, if Connection Master or some other non 4.1SC compatible program is being used, it may be necessary for the application to enable SCbus streams.

The Multi-Chassis Board is limited to, at most, eight transmit streams, on the SCbus. Each stream is comprised of 64 timeslots. The streams are selected using the **xds_sc_select** function. Streams should be selected so that no two Boards are enabled to transmit on the same SCbus timeslots. It is the responsibility of the application to manage stream selection.
This function uses a character array of eight entries, each of which should be in the range ‘0’ to ‘2’. The first entry is used to select either stream 0 or 8, the second to select 1 or 9, and so on. An entry of ‘0’ will disable transmitting on either of the streams for that position.

Example

xds_sbx_select(1, “11110000”);	This will enable board 1 to transmit on streams 0-3 or timeslots 0-255.

xds_scx_connect

xds_scx_connect(board_number, port, sc_rcv_timeslot, scx_xmt_timeslot, scx_rcv_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int port;	a value between 0 and the maximum number of ports on the board indicating which port is outputting to the SCbus. Ports on the Multi-chassis Board are actually virtual entities related to switching between the SCx and SC buses.
int sc_rcv_timeslot;	a value between 0-1023 indicating which timeslot on the SCbus is serving as an input
int scx_xmt_timeslot;	a value between 0-1535 indicating which timeslot on the SCxbus is being output to
int scx_rcv_timeslot;	a value between 0-1535 indicating which timeslot on the SCxbus is serving as an input

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function creates a two-way connection between the SCbus and the SCxbus. The SCxbus can be used to connect multiple chassis.

Message Sent
“XCxxxiiiyyyzzzz” where xxx is the SCbus output port, **iii** is the SCbus input timeslot, **yyy** is the SCxbus output timeslot and **zzzz** is the SCxbus input timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

The SCxbus consists of 24 streams of 64 timeslots each for a total of 1536 timeslots. Because of hardware limitations, a Multi-Chassis Board can transmit on at most 8 of those streams or 512 timeslots. It can receive any of the 1536 timeslots. The **xds_scx_connect** function is used to create a full-duplex connection between the local SCbus and the SCxbus. The SCxbus transmit timeslots must be within the streams enabled for the board (see **xds_scx_select).** The Multi-Chassis Board at the other end of the connection must have a complementary connection established.

The choice of SCxbus timeslots is beyond the scope of the SC 4.1 specification and it is up to the application to select appropriate streams for transmitting on and to choose the specific timeslots to be used for each connection.

The Multi-Chassis Board may have as many as 512 virtual ports which can output to the local SCbus. Each virtual port is capable of making a connection from the SCxbus to the SCbus. The exact number of ports available is determined during initialization. It is possible to reserve fewer than the maximum number of 512 when an application or configuration does not require that many. The SCbus timeslot used as an output by a virtual port is determined by the port number and the base timeslot set for the board following the SC 4.1 specification. It may be determined by calling the function **xds_xmt_timeslot.**

Example

xds_scx_connect(1, 2, 1022, 512, 1535);	this creates a connection between timeslot 1535 on the SCxbus and port 2 on the SCbus, and between timeslot 1022 on the SCbus and timeslot 512 on the SCxbus.

xds_scx_listen

xds_scx_listen(board_number, port, scx_rcv_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int port	a value between 0 and the maximum number of ports on the board indicating which port is outputting to the SCbus. Ports on the Multi-chassis Board are actually virtual entities related to switching between the SCx and SC buses.
int scx_rcv_timeslot;	a value between 0-1535 indicating which timeslot on the SCxbus is serving as an input

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function creates a one-way audio connection from the SCxbus timeslot to the SCbus port.

Message Sent
"XLIxxxzzzz" where **xxx** is the SCbus port and **zzzz** is the SCxbus timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was used
ILL_PORT	an invalid port was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function performs essentially half of an **xds_scx_connect**. Any of the 1536 SCxbus timeslots may be connected to a port on the SCbus using this function. See **xds_scx_connect** for details of the SCxbus.

The Multi-Chassis Board may have as many as 512 virtual ports which can output to the local SCbus. Each virtual port is capable of making a connection from the SCxbus to the SCbus. The exact number of ports available is determined during initialization. It is possible to reserve fewer than the maximum number of 512 when an application or configuration does not require that many ports. The SCbus timeslot used as an output by a virtual port is determined by the port number and the base timeslot set for the board following the 4.1 SC specification. It may be determined by calling the function **xds_xmt_timeslot.**

Example

xds_scx_listen(1, 2, 1535);	this creates an audio path between timeslot 1535 on the SCxbus and port 2 on the SCbus using board 1.

xds_scx_pattern

xds_scx_pattern(board_number, scx_xmt_timeslot, pattern);

unsigned char board_number;	a value indicating which board the command is for
int scx_xmt_timeslot	a value between 0-1535 indicating which timeslot on the SCxbus is being output to
unsigned char pattern;	a value between 0x0 and 0xFF to be output to the SCxbus

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function may be used to output a fixed pattern on the SCxbus. This may be used for diagnostic purposes or to place a "silence" pattern on the SCxbus.

Message Sent
"XPOyyypp" where **yyy** is the SCxbus timeslot and **pp** is the pattern value.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The SCxbus timeslot for the pattern to be output on must be on a stream that has been selected by this board (see **xds_scx_select).** It is also necessary to enable the SCxbus pattern capability using the **xds_scx_pattern_ena** function. Enabling this capability disables the ability to make connections from stream 15 of the SCbus to the SCxbus. This affects timeslots 960-1023 on the SCbus. Patterns may be output on a maximum of 64 SCxbus timeslots at a time. A pattern, once established may be disabled by using the **xds_scx_rls** function, or by disabling all SCxbus patterns with the **xds_scx_pattern_ena** function.

Example

xds_scx_pattern(1, 512, 0x45);	this outputs the pattern 0x45 on timeslot 512 of the SCxbus from board 1.

xds_scx_pattern_ena

xds_scx_pattern_ena(board_number, ena_dis);

unsigned char board_number;	a value indicating which board the command is for
char ena_dis;	a character to indicate whether to enable the SCxbus pattern output hardware, an ‘E’ enables and a ‘D’ disables

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function is used to enable and disable the hardware used to generate a pattern on the SCxbus.

Message Sent
“SPE” for enable or “SPD” for disable.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an invalid pattern mode was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to enable or disable the hardware used to generate a pattern on the SCxbus using the **xds_scx_pattern** function. When enabled, timeslots 960-1023 on the SCbus can not be connected to the SCxbus.

Example

xds_scx_pattern_ena(1, ‘E’);	this will enable the SCxbus pattern hardware.
xds_scx_pattern_ena(1, ‘D’);	this will disable the SCxbus pattern hardware.

xds_scx_rls

xds_scx_rls(board_number, scx_xmt_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int scx_xmt_timeslot;	a value between 0-1535 indicating which timeslot on the SCxbus is being output to

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function will release a connection between the SCbus and the SCxbus and disable the output on the SCxbus timeslot. This function will break down only half of a full duplex connection.

Message Sent
"XDOyyy" where **yyy** is the SCxbus transmit timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The **xds_scx_rls** function releases a connection from the SCbus to the SCxbus that has been established using either the **xds_scx_connect** or **xds_scx_transmit** functions. To release a connection from the SCxbus to the SCbus, the **xds_sc_rls** function must be used. Therefore, to release a full duplex connection both the **xds_sc_rls** and **xds_scx_rls** functions must be used.

Example

xds_scx_rls(1, 512);	this will disable the output from board 1 on timeslot 512 of the SCxbus.

xds_scx_select

xds_scx_select(board_number, streams);

unsigned char board_number;	a value indicating which board the command is for
char *streams;	a pointer to a character array indicating which SCxbus streams to enable

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function is used to select which SCxbus streams the Multi-Chassis board can transmit on. The board can transmit on, at most, eight streams.

Message Sent
“SHabcdefgh” where **abcdefgh** is the stream selection information.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The Multi-Chassis Board is limited to, at most, eight transmit streams. Each stream is comprised of 64 timeslots. The streams are selected using the **xds_scx_select** function. Streams should be selected so that no two Multi-Chassis Boards are enabled to transmit on the same SCxbus stream. It is the responsibility of the application to manage stream selection.

This function uses a character array of eight entries, each of which should be in the range ‘0’ to ‘3’. The first entry is used to select either stream 0, 8, or 16, the second to select 1, 9 or 17, and so on. An entry of ‘0’ will disable transmitting on any of the three streams for that position. Note, that if more than three chassis are in a system, eight transmit streams will not be available for each chassis.

Example

xds_scx_select(1, “11110000”);	this will enable board 1 to transmit on streams 0-3 or timeslots 0-255.

xds_scx_transmit

xds_scx_transmit(board_number, scx_xmt_timeslot, sc_rcv_timeslot);

unsigned char board_number;	a value indicating which board the command is for
int scx_xmt_timeslot;	a value between 0-1535 indicating which timeslot on the SCxbus is serving as an output
int sc_rcv_timeslot;	a value between 0-1023 indicating which timeslot on the SCbus is serving as an input

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function creates a one-way audio connection between a timeslot on the SCbus and a timeslot on the SCxbus.

Message Sent
“XLOyyyiii” where **iii** is the SCbus input timeslot and **yyy** is the SCxbus output timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The SCxbus consists of 24 streams of 64 timeslots each for a total of 1536 timeslots. A Multi-Chassis Board can transmit on at most 8 of those streams or 512 timeslots. It can receive any of the 1536 timeslots. The **xds_scx_transmit** function is used to create a half-duplex connection between the local SCbus and the SCxbus. The SCxbus transmit timeslots must be within those enabled for the board (see **xds_scx_select).** The Multi-Chassis Board at the other end of the connection must have a complementary connection established. As many as 20 Multi-Chassis Boards may be connected to the SCxbus, each with one stream reserved for transmitting.
The choice of SCxbus timeslots is beyond the scope of the 4.1 SC specification and it is up to the application to select appropriate streams for transmitting on and to choose the specific timeslots to be used for each connection.

Example

xds_scx_transmit(1, 512, 1022);	this creates an audio path from timeslot 1022 on the SCbus on board 1 to timeslot 512 on the SCxbus.

xds_scx_set_clock

xds_scx_set_clock(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	a value between 0x0 and 0xA selecting a clock mode for the SCxbus

Applicable boards
XDS SCSA Multi-Chassis Board

Purpose
This function is used to set the clock mode for the SCxbus. Because the clocks for the SCxbus and SCbus are related, this function may affect the SCbus clocks.

Message Sent
"SCx" where **x** is the clock mode.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid clock mode was used
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments

In a system of PCs tied together using the SCxbus and Multi-Chassis Boards, there must be only one source of the master clock. This clock may be generated by any board in the system and may be created by a local oscillator on that board or derived from an outside source such as a T1 span. All other boards in the system must be slaved to this clock. The Multi-Chassis Board in the same chassis with the master clock must derive the SCxbus clock from the SCbus and generate the clocks for the SCxbus. Other Multi-Chassis Boards in the system must derive their clocks from the SCxbus and supply the clocks to the SCbus in their chassis. It is possible for a Multi-Chassis Board to supply all the clocks to both buses from an on board oscillator.

Because all boards must be set to compatible clock modes, this function must be used with care. In particular, because the SCbus master clock sources are set at initialization under the 4.1 SC specification, care must be taken to avoid changing the clocks in relation to the SCbus when changing the clock mode. In other words, if the board is supplying the clock to the SCbus, only those modes that also supply the clock to the SCbus should be chosen.

Alternately, if the SCbus is supplying the clock, only those modes where the SCbus also supplies the clock should be chosen. A more detailed discussion of the clocks may be found in the manual on the XDS SCSA Multi-Chassis Interface Board.

The possible clock modes for the Multi-Chassis Boards are:

0 - no clock to or from the bus
1 - SCbus provides the clock, clock is supplied to the SCxbus right clock
2 - SCbus provides the clock, clock is supplied to the SCxbus left clock
3 - the board supplies the clock to the SCbus and the SCxbus right clock
4 - the board supplies the clock to the SCbus and the SCxbus left clock
5 - the clock is derived from 8KREF, clock is supplied to the SCbus and SCxbus right clock
6 - the clock is derived from 8KREF, clock is supplied to the SCbus and SCxbus left clock
7 - SCxbus right clock provides clock, clock is supplied to the SCbus
8 - SCxbus right clock provides clock, clock is supplied to the SCbus and SCxbus left clock
9 - SCxbus left clock provides clock, clock is supplied to the SCbus
10 - SCxbus left clock provides clock, clock is supplied to the SCxbus right clock

The board will respond with a message of type 2, subtype 3 when the clock mode has been set. If a clock mode greater than 10 is chosen the board will respond with an error message of type 4, subtype 3.

Example

xds_scx_set_clock(1, 2);	this will set board 1 to derive the clock from the SCbus and provide the clock to the SCxbus left clock.

This page was intentionally left blank.

SCSA/MVIP Bridge Board Functions

This page was intentionally left blank.

xds_xb_connect

int xds_xb_connect(board_number, timeslot_1, timeslot_2);

unsigned char board_number;	a value indicating which board the command is for
int timeslot_1;	a value between 0-1535 indicating one of the timeslots being connected
int timeslot_2;	a value between 0-1535 indicating the other timeslot being connected

Applicable boards
XDS SCSA/MVIP Bridge Board

Purpose
This function creates a two-way connection between the SCbus and the MVIP bus. This can be used to connect MVIP boards to SCbus boards. It can also be used to make connections between timeslots on the same bus.

Message Sent
"CCxxxxyyyy" where **xxxx** is one of the timeslots, and **yyyy** is the other timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The SCbus consists of 16 streams of 64 timeslots each for a total of 1024 timeslots. The MVIP bus consists of 16 streams of 32 timeslots each for a total of 512 timeslots. In the timeslot arguments, SCbus timeslots occupy the range 0-1023 and MVIP timeslots occupy the range 1024-1535. Note that on the MVIP bus streams 0-7 are labeled DSo0-DSo7, and streams 8-15 are labeled DSi0-DSi7. The DSo and DSi streams form a pair with a timeslot on one stream serving as the input timeslot and the corresponding timeslot on the other stream serving as the output timeslot. Specifying stream 1 means inputting on DSo1 or stream 1 and outputting on DSi1 or stream 9.

A Bridge Board can transmit and receive on at most 4 of the SCbus streams or 256 timeslots in each direction. The **xds_xb_connect** function is used to create a full-duplex connection between the SCbus and the MVIP bus. The SCbus transmit and receive timeslots must be within the streams enabled for the board (see **xds_xbx_select, xds_xbr_select).** The actual SCbus timeslots used will depend on which streams are enabled. The transmit and receive timeslots on both the MVIP and the SCbus are assumed to be paired, with a single timeslot argument specifying both a transmit and receive timeslot. The SCbus argument specifies the transmit timeslot, with the the receive timeslot being the corresponding timeslot on the streams specified by **xds_xbr_select**. The choice of SCbus timeslots is governed by the SC 4.1 specification. Because of the way that the Bridge Board is used it is up to the application to select appropriate streams for transmitting on and to choose the specific timeslots to be used for each connection.

Example

xds_xb_connect(1, 355, 1073);	this creates a connection to timeslot 355 on the SCbus from timeslot 17 on stream 1 of the MVIP bus, and from timeslot 99 on the SCbus to timeslot 17 on stream 9 on the MVIP bus. This assumes that the streams 0-3 (timeslots 0 - 255) on the SCbus have been enabled for receiving and streams 4-7 (timeslots 256-511) have been enabled for transmitting.

Timeslot 355 = 256 + 99 = 256 + (1 x 64) + 17 on the SCbus or transmit on timeslot 17 of stream 5 and receive on timeslot 17 of stream 1.

Timeslot 1073 = 1024 + (1 x 32) + 17 on the MVIP bus or receive on timeslot 17 of stream 1 and transmit on timeslot 17 stream 9 (streams 1 & 9 are paired) on the MVIP bus

xds_xb_connect_2

int xds_xb_connect_2(board_number, timeslot_1, timeslot_2, timeslot_3);

unsigned char board_number;	a value indicating which board the command is for
int timeslot_1;	a value between 0-1535 indicating a transmit and receive pair of the timeslots on one of the busses
int timeslot_2;	a value between 0-1535 indicating the source timeslot
int timeslot_3;	a value between 0-1535 indicating the destination timeslot

Applicable boards
XDS SCSA/MVIP Bridge Board

Purpose
This function creates a two-way connection between the MVIP bus and the SCbus where the timeslots on one of the busses do not form a transmit/receive pair. It is intended to make a full-duplex connection where the MVIP timeslots are specified by one argument and the SCbus timeslots are not related.

Message Sent
"CSxxxxyyyyzzzz" where **xxxx** specifies a transmit/receive pair of timeslots, **yyyy** is the source timeslot and **zzzz** is the destination timeslot of the other half of the connection.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The SCbus consists of 16 streams of 64 timeslots each for a total of 1024 timeslots. The MVIP bus consists of 16 streams of 32 timeslots each for a total of 512 timeslots. In the timeslot arguments, SCbus timeslots occupy the range 0-1023 and MVIP timeslots occupy the range 1024-1535. Note that on the MVIP bus streams 0-7 are labeled DSo0-DSo7, and streams 8-15 are labeled DSi0-DSi7. The DSo and DSi streams form a pair with a timeslot on one stream serving as the input timeslot and the corresponding timeslot on the other stream serving as the output timeslot. Specifying stream 1 means inputting on DSo1 or stream 1 and outputting on DSi1 or stream 9.

A Bridge Board can transmit and receive on at most 4 of the SCbus streams or 512 timeslots. The **xds_xb_connect_2** function is used to create a full-duplex connection between the SCbus and the MVIP bus. The SCbus transmit and receive timeslots must be within the streams enabled for the board (see **xds_xbx_select, xds_xbr_select).** The actual SCbus timeslots used will depend on which streams are enabled.

The transmit and receive timeslots specified by the first argument are assumed to be paired, with a single timeslot argument specifying both a transmit and receive timeslot, while the second and third timeslot arguments specify an unpaired receive and transmit timeslot on the opposite bus. Typically the first timeslot will be on the MVIP bus where such pairings are normal while the second and third arguments specify timeslots on the SCbus.

The choice of SCbus timeslots is governed by the SC 4.1 specification. Because of the way that the Bridge Board is used it is up to the application to select appropriate streams for transmitting on and to choose the specific timeslots to be used for each connection.

Example

xds_xb_connect_2(1, 1073, 31, 355);	this creates a connection to timeslot 355 on the SCbus from timeslot 17 on stream 1 of the MVIP bus, and from timeslot 31 on the SCbus to timeslot 17 on stream 9 on the MVIP bus. This assumes that the streams 0-3 (timeslots 0-255) on the SCbus have been enabled for receiving and streams 4-7 (timeslots 256-511) have been enabled for transmitting.

Timeslot 1073 = 1024 + (1 x 32) + 17 on the MVIP bus or receive on timeslot 17 of stream 1 and transmit on timeslot 17 stream 9 (streams 1 & 9 are paired on the MVIP bus.

Source timeslot 31 = (0 x 64) + 31 on the SCbus or timeslot 31 of stream 0.

Destination timeslot 355 = 256 + 99 = 256 + (1 x 64) + 17 = (5 x 64) + 17 on the SCbus or timeslot 17 of stream 5.

xds_xb_disconnect

int xds_xb_disconnect(board_number, timeslot);

unsigned char board_number;	a value indicating which board the command is for
int timeslot;	a value between 0-1535 indicating the timeslot being disconnected

Applicable boards
XDS SCSA/MVIP Bridge Board

Purpose
This function disables a connection between the SCbus and the MVIP bus.

Message Sent
"CDxxxx" where **xxxx** is the timeslot

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This function is used to undo connections made using the **xds_xb_connect, xds_xb_listen, xds_xb_pattern,** and **xds_xb_connect_2** functions. The timeslot argument can refer to either the SCbus or the MVIP bus. The function only disables the output to the specified timeslot. Therefore, it is necessary to invoke the function twice for a full duplex connection.

Example

xds_xb_disconnect(1, 355);	
xds_xb_disconnnct(1, 1073);	this disables output to timeslot 355 on the SCbus and to timeslot 17 on stream 9 on the MVIP bus. This assumes that the streams 4-7 have been enabled for transmitting.

Timeslot 355 = 256 + 99 = 256 + (1 x 64) + 17 = (5 x 64) + 17 on the SCbus or timeslot 17 of stream 5.

Timeslot 1073 = 1024 + (1 x 32) + 17 on the MVIP bus or timeslot 17 of stream 9.

xds_xb_listen

int xds_xb_listen(board_number, timeslot_1, timeslot_2);

char board_number;	a value indicating which board the command is for
int timeslot_1;	a value between 0-1535 indicating the destination timeslot
int timeslot_2;	a value between 0-1535 indicating the source timeslot

Applicable boards
XDS SCSA/MVIP Bridge Board

Purpose
This function creates a one-way connection between the SCbus and the MVIP bus or between the MVIP bus and the SCbus. This can be used to connect MVIP boards to SCbus boards. It can also be used to make connections between timeslots on the same bus.

Message Sent
“CLxxxxyyyy” where **xxxx** is the destination timeslot, and **yyyy** is the source timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The SCbus consists of 16 streams of 64 timeslots each for a total of 1024 timeslots. The MVIP bus consists of 16 streams of 32 timeslots each for a total of 512 timeslots. In the timeslot arguments, SCbus timeslots occupy the range 0-1023 and MVIP timeslots occupy the range 1024-1535. Note that on the MVIP bus streams 0-7 are labeled DSo0-DSo7, and streams 8-15 are labeled DSi0-DSi7. The DSo and DSi streams form a pair with a timeslot on one stream serving as the input timeslot and the corresponding timeslot on the other stream serving as the output timeslot. Specifying stream 1 means inputting on DSo1 or stream 1 and outputting on DSi1 or stream 9.

A Bridge Board can transmit and receive on at most 4 of the SCbus streams or 256 timeslots in each direction. The **xds_xb_listen** function is used to create a half-duplex connection between the two busses. The SCbus transmit and receive timeslots must be within the streams enabled for the board (see **xds_xbx_select, xds_xbr_select).** The actual SCbus timeslots used will depend on which streams are enabled.

The choice of SCbus timeslots is governed by the SC 4.1 specification. Because of the way that the Bridge Board is used it is up to the application to select appropriate streams for transmitting on and to choose the specific timeslots to be used for each connection.

Example
xds_xb_listen(1, 355, 1073); this creates a connection to timeslot 355 on the SCbus from timeslot 17 on stream 1 of the MVIP bus. This assumes that streams 4-7 on the SCbus have been enabled for transmitting.

Timeslot 355 = 256 + 99 = 256 + (1 x 64) + 17 = (5 x 64) + 17 on the SCbus or timeslot 17 of stream 5.

xds_xb_pattern

int xds_xb_pattern(board_number, timeslot_1, pattern);

unsigned char board_number;	a value indicating which board the command is for
int timeslot_1;	a value between 0-1535 indicating the output timeslot
unsigned char pattern;	a value between 0x00 and 0xFF to be output

Applicable boards
XDS SCSA/MVIP Bridge Board

Purpose
This function may be used to output a fixed pattern to either the SCbus or the MVIP bus. This may be used for diagnostic purposes or to output "silence" or some other pattern on a particular timeslot.

Message Sent
"CPxxxxpp" where **xxxx** is the output timeslot, and **pp** is the value to be output

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_SLOT	an invalid timeslot was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The SCbus consists of 16 streams of 64 timeslots each for a total of 1024 timeslots. The MVIP bus consists of 16 streams of 32 timeslots each for a total of 512 timeslots. In the timeslot arguments, SCbus timeslots occupy the range 0-1023 and MVIP timeslots occupy the range 1024-1535. Note that on the MVIP bus streams 0-7 are labeled DSo0-DSo7, and streams 8-15 are labeled DSi0-DSi7. The DSo and DSi streams form a pair with a timeslot on one stream serving as the input timeslot and the corresponding timeslot on the other stream serving as the output timeslot. Specifying stream 1 means inputting on DSo1 or stream 1 and outputting on DSi1 or stream 9.

A Bridge Board can transmit on at most 4 of the SCbus streams or 256 timeslots in each direction. The **xds_xb_pattern** function is used to output a pattern on either the SCbus and the MVIP bus. The SCbus transmit timeslots must be within the streams enabled for the board (see **xds_xbx_select).** The actual SCbus timeslots used will depend on which streams are enabled. The choice of SCbus timeslots is governed by the SC 4.1 specification. Because of the way that the Bridge Board is used it is up to the application to select appropriate streams for transmitting on and to choose the specific timeslots to be used for each connection.

Example

xds_xb_pattern(1, 355, 0xFF);	this causes 0xFF to be output on timeslot 355 on the SCbus. This assumes that streams 4-7 on the SCbus have been enabled for transmitting.

Timeslot 355 = 256 + 99 = 256 + (1 x 64) + 17 = (5 x 64) + 17 on the SCbus or timeslot 17 of stream 5.

xds_xbr_select

xds_xbr_select(board_number, streams);

unsigned char board_number;	a value indicating which board the command is for
char *streams;	a pointer to a character array indicating which SCbus streams to enable for receiving

Applicable boards
XDS SCSA/MVIP Bridge Board

Purpose
This function is used to select which SCbus streams the Bridge board can receive from. Hardware limitations restrict the board to receiving from at most four streams.

Message Sent
"SBRabcd" where **abcd** is the stream selection information.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The SCSA/MVIP Board is limited to, at most, four receive streams. Each stream is comprised of 64 timeslots. The streams are selected using the **xds_xbr_select** function.
This function uses a character array of four entries, each of which should be in the range '0' to '3'. The first entry is used to select either stream 0, 4, 8, or 12, the second to select 1, 5, 9 or 13, and so on. An entry of '0' will disable receiving from any of the four streams for that position.

Example

xds_xbr_select(1, "1111");	this will enable board 1 to receive from streams 0-3 or timeslots 0-255.

xds_xbx_select

xds_xbx_select(board_number, streams);

unsigned char board_number;	a value indicating which board the command is for
char *streams;	a pointer to a character array indicating which SCbus streams to enable for transmitting

Applicable boards
XDS SCSA/MVIP Bridge Board

Purpose
This function is used to select which SCbus streams the Bridge board can transmit to. Hardware limitations restrict the board to transmitting on at most four streams.

Message Sent
“SBXabcd” where **abcd** is the stream selection information.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
The SCSA/MVIP Board is limited to, at most, four transmit streams. Each stream is comprised of 64 timeslots. The streams are selected using the **xds_xbx_select** function. Streams should be selected so that the SCSA/MVIP Bridge Board is not enabled to transmit on the same SCbus stream as any other board. It is the responsibility of the application to manage stream selection. This function uses a character array of four entries, each of which should be in the range ‘0’ to ‘3’. The first entry is used to select either stream 0, 4, 8, or 12, the second to select 1, 5, 9 or 13, and so on. An entry of ‘0’ will disable transmitting on any of the four streams for that position.

Example

xds_xbr_select(1, “2222”);	this will enable board 1 to transmit on streams 4-7 or timeslots 256-511.

This page was intentionally left blank.

Voice Playback Functions

This page was intentionally left blank.

xds_voice_play_file

xds_voice_play_file(pPlayFileParm, Callback_func);

typedef struct PLAYFILEPARM {

unsigned char	board_number;	The board number of an XDS T1/E1 board that was used for voice playback.
unsigned char	voice_channel;	The voice channel to be used on the T1/E1 board to play the file. The value will be 0 – 3 for the 4 port boards and 0 – 7 for the 8 port boards.
short	repeat_number;	How many times the user wants to play the voice file. It can be infinite by using the XDS_REPEAT_INFINITY (-1) value.
unsigned char	dtmf_terminate;	This will control the voice playback termination behavior. If this value equals 1, it will stop playing when it receives the first DTMF tone. If this value equals 0, it will not stop playing when it receives first DTMF tone. DTMF detection must be enabled in order to terminate on DTMF.
unsigned char	dtmf_detect;	This parameter will set the DTMF detection mode. If this value is 1, it will report each DTMF tone detected. If this value is 0, it will not report any DTMF tones detected.
unsigned char	*FileName;	A pointer to the name of the file to be played.
short	file_type;	The audio type selected for play. These can be: XDS_TYPE_WAVE, XDS_TYPE_RAW_ALAW, XDS_TYPE_RAW_ULAW, XDS_TYPE_RAW_ADPCM.

} PlayFileParm, *pPlayFileParm;

typedef struct CallbackParm {

unsigned char	board_number;	The board number that was used to play the file.
unsigned char	voice_channel;	The channel number of the channel that played the file.
int	callback_code;	The callback reports a callback code, which can be can be XDS_DONE_PLAY, XDS_USER_STOPPED, XDS_DTMF_TERMINATED, or XDS_RECEIVED_DTMF.
unsigned char	dtmf_tone;	The DTMF tone received by the board, can be '0' – '9', '*', '#', or '@' (end of playback reached).

} CallbackParms, *pCallbackParms;

Applicable boards
Infinity Series H.100 T1/E1 boards.

Purpose
This function is used for voice playback from a supported audio file.

Messages Sent

"PV0dstc"	-	set voice playback options, where **d** is the voice channel, **s** is the stop on DTMF mode, **t** is the DTMF detection mode, and **c** is the data coding format
"SVE"	-	to enable the voice playback on the board

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

XDS_WRONG_PARAMETER	– invalid voice file parameters were used
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than the T1/E1 board was selected
XDS_SYSTEM_ERROR	– system memory for the board parameters was unable to be allocated
XDS_WRONG_VOICE_CHANNEL	– an invalid voice channel was selected for use
XDS_BAD_FILE	**–** voice file other than supported wav formats was chosen for play
IOCTL	- an IOCTL was called and returns a return value from the IOCTL call

Comments
At the time of this manual's release, voice playback is only available on the XDS H.100 T1/E1 board and must have the current firmware loaded on the DSP and processor.

Supported audio playback formats at this time include raw A-law, raw μ-law, raw ADPCM, and A-law & μ-law in .wav format.

Example

xds_voice_play_file((void *)&parm, &Complete_function);

this will play a voice file back with the parameters set in the *parm* data structure and use the function *Complete_function()* as the callback function.

xds_voice_play_buffer

xds_voice_play_buffer(pPlayBufParm, Callback_func);

typedef struct PLAYBUFPARM {

unsigned char	board_number;	The board number of an XDS T1/E1 board that was used for voice playback.
unsigned char	voice_channel;	The voice channel to be used on the T1/E1 board to play the file. The value will be 0 – 3 for the 4 port boards and 0 – 7 for the 8 port boards.
short	repeat_number;	How many times the user wants to play the voice file. It can be infinite by using the XDS_REPEAT_INFINITY (-1) value.
unsigned char	dtmf_terminate;	This will control the voice playback termination behavior. If this value equals 1, it will stop playing when it receives the first DTMF tone. If this value equals 0, it will not stop playing when it receives first DTMF tone. DTMF detection must be enabled in order to terminate on DTMF.
unsigned char	dtmf_detect;	This parameter will set the DTMF detection mode. If this value is 1, it will report each DTMF tone detected. If this value is 0, it will not report any DTMF tones detected.
unsigned char	*pBuffer;	A pointer to a memory buffer with a voice file in its contents.
short	data_type;	The audio type selected for play. These can be: XDS_TYPE_WAVE, XDS_TYPE_RAW_ALAW, XDS_TYPE_RAW_ULAW, XDS_TYPE_RAW_ADPCM.
unsigned long	length;	The length of an audio file buffered.

} PlayBufParm, *pPlayBufParm;

typedef struct CallbackParm {

unsigned char	board_number;	The board number that was used to play the file.
unsigned char	voice_channel;	The channel number of the channel that played the file.
int	callback_code;	The callback reports a callback code, which can be can be XDS_DONE_PLAY, XDS_USER_STOPPED, XDS_DTMF_TERMINATED, or XDS_RECEIVED_DTMF.
unsigned char	dtmf_tone;	The DTMF tone received by the board, can be ‘0’ – ‘9’, ‘*’, ‘#’, or ‘@’ (end of playback reached).

} CallbackParms, *pCallbackParms;

Applicable boards
Infinity Series H.100 T1/E1 boards.

Purpose
This function is used for voice playback from a memory buffer.

Messages Sent

"PV0dstc"	-	set voice playback options, where **d** is the voice channel, **s** is the stop on DTMF mode, **t** is the DTMF detection mode, and **c** is the data coding format
"SVE"	-	to enable the voice playback on the board

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

XDS_WRONG_PARAMETER	– invalid voice file parameters were used or memory buffer is NULL
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than the T1/E1 board was selected
XDS_SYSTEM_ERROR	– system memory for the board parameters was unable to be allocated
XDS_WRONG_VOICE_CHANNEL	– an invalid voice channel was selected for use
XDS_BAD_FILE	**–** voice file other than supported wav formats was chosen for play
IOCTL	- an IOCTL was called and returns a return value from the IOCTL call

Comments
At the time of this manual's release, voice playback is only available on the XDS H.100 T1/E1 board and must have the current firmware loaded on the DSP and processor.

Supported audio playback formats at this time include raw A-law, raw μ-law, raw ADPCM, and A-law & μ-law in .wav format.

Example

xds_voice_play_buffer((void *)&parm, &Complete_function);

this will play a voice file back with the parameters set in the *parm* data structure and use the function *Complete_function()* as the callback function.

xds_voice_play_file_index

xds_voice_play_file_index(pPlayFileIndexParm, Callback_func);

typedef struct PLAYFILEINDEXPARM {

unsigned char	board_number;	The board number of an XDS T1/E1 board that was used for voice playback.
unsigned char	voice_channel;	The voice channel to be used on the T1/E1 board to play the file. The value will be 0 – 3 for the 4 port boards and 0 – 7 for the 8 port boards.
short	repeat_number;	How many times the user wants to play the indexed file list. It can be infinite by using the XDS_REPEAT_INFINITY (-1) value.
unsigned char	dtmf_terminate;	This will control the voice playback termination behavior. If this value equals 1, it will stop playing when it receives the first DTMF tone. If this value equals 0, it will not stop playing when it receives first DTMF tone. DTMF detection must be enabled in order to terminate on DTMF.
unsigned char	dtmf_detect;	This parameter will set the DTMF detection mode. If this value is 1, it will report each DTMF tone detected. If this value is 0, it will not report any DTMF tones detected.
char	**pFiles;	A pointer to a pointer of a file list (array) that holds the files to be played. The last position in this list must be NULL.
short	file_type;	The audio type selected for play. These can be: XDS_TYPE_WAVE, XDS_TYPE_RAW_ALAW, XDS_TYPE_RAW_ULAW, XDS_TYPE_RAW_ADPCM.

} PlayFileIndexParm, *pPlayFileIndexParm;

typedef struct CallbackParm {

unsigned char	board_number;	The board number that was used to play the file.
unsigned char	voice_channel;	The channel number of the channel that played the file.
int	callback_code;	The callback reports a callback code, which can be can be XDS_DONE_PLAY, XDS_USER_STOPPED, XDS_DTMF_TERMINATED, or XDS_RECEIVED_DTMF.
unsigned char	dtmf_tone;	The DTMF tone received by the board, can be ‘0’ – ‘9’, ‘*’, ‘#’, or ‘@’ (end of playback reached).

} CallbackParms, *pCallbackParms;

Applicable boards
Infinity Series H.100 T1/E1 boards.

Purpose
This function is used for voice playback from an index of supported audio files.

Messages Sent

"PV0dstc"	-	set voice playback options, where **d** is the voice channel, **s** is the stop on DTMF mode, **t** is the DTMF detection mode, and **c** is the data coding format
"SVE"	-	to enable the voice playback on the board

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

XDS_WRONG_PARAMETER	– invalid voice file parameters were used
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than the T1/E1 board was selected
XDS_SYSTEM_ERROR	– system memory for the board parameters was unable to be allocated
XDS_WRONG_VOICE_CHANNEL	– an invalid voice channel was selected for use
XDS_BAD_FILE	**–** voice file other than supported wav formats was chosen for play
XDS_FILES_NOT_TERMINATED	**–** buffer list is not terminated properly
IOCTL	- an IOCTL was called and returns a return value from the IOCTL call

Comments
At the time of this manual's release, voice playback is only available on the XDS H.100 T1/E1 board and must have the current firmware loaded on the DSP and processor.

Supported audio playback formats at this time include raw A-law, raw μ-law, raw ADPCM, and A-law & μ-law in .wav format.

Example

xds_voice_play_file_index((void *)&parm, &Complete_function);

this will play an index of voice files back with the parameters set in the *parm* data structure and use the function *Complete_function()* as the callback function.

xds_voice_play_buffer_index

xds_voice_play_buffer_index(pPlayBufIndexParm, Callback_func);

typedef struct PLAYBUFINDEXPARM {

unsigned char	board_number;	The board number of an XDS T1/E1 board that was used for voice playback.
unsigned char	voice_channel;	The voice channel to be used on the T1/E1 board to play the file. The value will be 0 – 3 for the 4 port boards and 0 – 7 for the 8 port boards.
short	repeat_number;	How many times the user wants to play the indexed buffer list. It can be infinite by using the XDS_REPEAT_INFINITY (-1) value.
unsigned char	dtmf_terminate;	This will control the voice playback termination behavior. If this value equals 1, it will stop playing when it receives the first DTMF tone. If this value equals 0, it will not stop playing when it receives first DTMF tone. DTMF detection must be enabled in order to terminate on DTMF.
unsigned char	dtmf_detect;	This parameter will set the DTMF detection mode. If this value is 1, it will report each DTMF tone detected. If this value is 0, it will not report any DTMF tones detected.
BufferInfo	**pBuffers;	A pointer to a pointer of a buffer list (array) that holds the buffers to be played. The last position in this list must be NULL.
short	data_type;	The audio type selected for play. These can be: XDS_TYPE_WAVE, XDS_TYPE_RAW_ALAW, XDS_TYPE_RAW_ULAW, XDS_TYPE_RAW_ADPCM.
unsigned long	length;	The length of an audio file buffered.

} PlayBufIndexParm, *pPlayBufIndexParm;

typedef struct CallbackParm {

unsigned char	board_number;	The board number that was used to play the file.
unsigned char	voice_channel;	The channel number of the channel that played the file.
int	callback_code;	The callback reports a callback code, which can be can be XDS_DONE_PLAY, XDS_USER_STOPPED, XDS_DTMF_TERMINATED, or XDS_RECEIVED_DTMF.
unsigned char	dtmf_tone;	The DTMF tone received by the board, can be ‘0’ – ‘9’, ‘*’, ‘#’, or ‘@’ (end of playback reached).

} CallbackParms, *pCallbackParms;

```
typedef struct BUFFERINFO
{
       unsigned char *pBuffer;          A pointer to a memory buffer with a voice file in its
                                        contents.
       unsigned long  length;           The length of an audio file buffered.
} BufferInfo, *pBufferInfo;
```

Applicable boards
Infinity Series H.100 T1/E1 boards.

Purpose
This function is used for voice playback from an index of buffered voice files.

Messages Sent

"PV0dstc"	-	set voice playback options, where **d** is the voice channel, **s** is the stop on DTMF mode, **t** is the DTMF detection mode, and **c** is the data coding format
"SVE"	-	to enable the voice playback on the board

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

XDS_WRONG_PARAMETER	– invalid voice file parameters were used or memory buffer is NULL
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than the T1/E1 board was selected
XDS_SYSTEM_ERROR	– system memory for the board parameters was unable to be allocated
XDS_WRONG_VOICE_CHANNEL	– an invalid voice channel was selected for use
XDS_BAD_FILE	**–** the voice file buffered is other than supported wav formats was chosen for play
XDS_FILES_NOT_TERMINATED	**–** buffer list is not terminated properly
IOCTL	- an IOCTL was called and returns a return value from the IOCTL call

Comments
At the time of this manual's release, voice playback is only available on the XDS H.100 T1/E1 board and must have the current firmware loaded on the DSP and processor.

Supported audio playback formats at this time include raw A-law, raw μ-law, raw ADPCM, and A-law & μ-law in .wav format.

Example

xds_voice_play_buffer_index((void *)&parm, &Complete_function);

this will play an index of buffered voice files back with the parameters set in the *parm* data structure and use the function *Complete_function()* as the callback function.

xds_voice_stop_play

xds_voice_stop_play(void *);

The parameter passed is a pointer to either the pPlayFileParm or the pPlayBufferParm data structure, whichever one was used to start the playback.

Applicable boards
Infinity Series H.100 T1/E1 boards.

Purpose
This function stops playback of a file or memory prompt and disables DTMF detection for the selected voice channel.

Message Sent
None

Returns
This function will return a 0 if not successful and 1 if successful.

Comments
This function call can be used to stop either voice file or voice buffer playback. The user will pass a pointer to the data structure used to start the play to this function to stop play.

Example

```
xds_voice_stop_play((void *)&parm);
```

this will stop play back with the parameters set in the *parm* data structure.

xds_voice_playback_connect

xds_voice_playback_connect(pConnectionParms);

typedef struct ConnectionParm {

unsigned char	board_number;	The board number of an XDS T1/E1 board that was used to play the file.
unsigned char	voice_channel;	The voice channel to be used on the T1/E1 board to play the file. The value will be 0 – 3 for the 4 port boards and 0 – 7 for the 8 port boards.
char	bus;	The bus used to connect the playback channel to. For the H.100 bus, the bus parameter is 'H'. For the local bus, the bus parameter is 'L'.
int	stream;	The stream used to connect the playback to.
int	timeslot;	The timeslot used to connect the playback to.

} ConnectionParms, *pConnectionParms;

Applicable boards
Infinity Series H.100 T1/E1 boards.

Purpose
This function is used to make the physical board connection from the selected voice channel to an output bus, stream, and timeslot.

Message Sent
"MObssttEbsstt" - where the first **b** is the output bus, the first **ss** is the output stream, and the first **tt** is the output timeslot.

the second **b** is 'L' for the local bus, the second **ss** is the appropriate stream for the voice channel selected calculated according to voice channel number, and the second **tt** is 0xF.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than the T1/E1 board was selected
XDS_WRONG_VOICE_CHANNEL	– an invalid voice channel was selected for use
XDS_WRONG_STREAM	– an invalid output stream was used for the output bus selected
xds_ct_mvip_output_enable()	- the output enable function was called and returns a value from that function

Comments
Voice channels 0 through 3 can connect to local bus streams 0 through 7 and any H.100 bus stream. Voice channels 4 through 7 can connect to local bus streams 8 through F and any H.100 bus stream. It is not possible to connect voice channels 0 through 3 directly to local bus streams 8 through F. Likewise, it is not possible to connect voice channels 4 through 7 directly to local bus streams 0 through 7.

Example

```
xds_voice_playback_connect((void *)&cparm);
```

this will connect a voice file playback channel to a user-specified output bus, stream, and timeslot with the parameters set in the *cparm* data structure

xds_voice_dtmf_connect

xds_voice_dtmf_connect(pConnectionParms);

typedef struct ConnectionParm {

unsigned char	board_number;	The board number of an XDS T1/E1 board that was used to play the file.
unsigned char	voice_channel;	The voice channel to be used on the T1/E1 board to play the file. The value will be 0 – 3 for the 4 port boards and 0 – 7 for the 8 port boards.
char	bus;	The bus the DTMF detector is connected to. For the H.100 bus, the bus parameter is 'H'. For the local bus, the bus parameter is 'L'.
int	stream;	The stream the DTMF detector is connected to.
int	timeslot;	The timeslot the DTMF detector is connected to.

} ConnectionParms, *pConnectionParms;

Applicable boards
Infinity Series H.100 T1/E1 boards.

Purpose
This function is used to make the physical board connection from the selected voice channel DTMF detector to an input bus, stream, and timeslot.

Message Sent

"MObssttEbsstt"	-	where the first **b** is 'L' for the local bus, the first **ss** is the appropriate stream for the voice channel DTMF detector selected calculated according to voice channel number, and the first **tt** is 0x1F. the second **b** is the input bus, the second **ss** is the input, and second **tt** is the input stream.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than the T1/E1 board was selected
XDS_WRONG_VOICE_CHANNEL	– an invalid voice channel was selected for use
XDS_WRONG_STREAM	– an invalid output stream was used for the output bus selected
xds_ct_mvip_output_enable()	- the output enable function was called and returns a value from that function

Comments
The DTMF detectors on voice channels 0 through 3 can connect to local bus streams 0 through 7 and any H.100 bus stream. DTMF detectors on voice channels 4 through 7 can connect to local bus streams 8 through F and any H.100 bus stream. It is not possible to connect the detectors for voice channels 0 through 3 directly to local bus streams 8 through F. Likewise, it is not possible to connect the detectors for voice channels 4 through 7 directly to local bus streams 0 through 7.

Example

```
xds_voice_dtmf_connect((void *)&cparm);
```

this will connect a voice channel DTMF detector to a user-specified input bus, stream, and timeslot with the parameters set in the *cparm* data structure

xds_voice_playback_disconnect

xds_voice_playback_disconnect(pConnectionParms);

typedef struct ConnectionParm {

unsigned char	board_number;	The board number of an XDS T1/E1 board that was used to play the file.
unsigned char	voice_channel;	The voice channel to be used on the T1/E1 board to play the file. The value will be 0 – 3 for the 4 port boards and 0 – 7 for the 8 port boards.
char	bus;	The bus used to connect the playback channel to. For the H.100 bus, the bus parameter is 'H'. For the local bus, the bus parameter is 'L'.
int	stream;	The stream used to connect the playback to.
int	timeslot;	The timeslot used to connect the playback to.

} ConnectionParms, *pConnectionParms;

Applicable boards
Infinity Series H.100 T1/E1 boards.

Purpose
This function is used to disconnect the physical board connection from the selected voice channel to an output bus, stream, and timeslot.

Message Sent

"MObssttD"	-	where the **b** is the output bus, the **ss** is the output stream, and the **tt** is the output timeslot.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than the T1/E1 board was selected
XDS_WRONG_VOICE_CHANNEL	– an invalid voice channel was selected for use
xds_ct_mvip_output_disable()	- the output enable function was called and returns a value from that function

Comments
The pointer to the data structure passed is the same as the one passed to the xds_voice_playback_connect() function.

Example
xds_voice_playback_disconnect((void *)&cparm);

this will disconnect a voice file playback channel to a user-specified output bus, stream, and timeslot with the parameters set in the *cparm* data structure

xds_voice_dtmf_disconnect

xds_voice_dtmf_disconnect(pConnectionParms);

typedef struct ConnectionParm {

unsigned char	board_number;	The board number of an XDS T1/E1 board that was used to play the file.
unsigned char	voice_channel;	The voice channel to be used on the T1/E1 board to play the file. The value will be 0 – 3 for the 4 port boards and 0 – 7 for the 8 port boards.
char	bus;	The bus the DTMF detector is connected to. For the H.100 bus, the bus parameter is 'H'. For the local bus, the bus parameter is 'L'.
int	stream;	The stream the DTMF detector is connected to.
int	timeslot;	The timeslot the DTMF detector is connected to.

} ConnectionParms, *pConnectionParms;

Applicable boards
Infinity Series H.100 T1/E1 boards.

Purpose
This function is used to make the physical board connection from the selected voice channel DTMF detector to an input bus, stream, and timeslot.

Message Sent

"MObssttEbsstt"	-	where the first **b** is 'L' for the local bus, the first **ss** is the appropriate stream for the voice channel DTMF detector selected calculated according to voice channel number, and the first **tt** is 0x1F. the second **b** is the input bus, the second **ss** is the input, and second **tt** is the input stream.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

xds_ct_mvip_output_disable()	- the output enable function was called and returns a value from that function

Comments
The pointer to the data structure passed is the same as the one passed to the xds_voice_dtmf_connect() function.

Example

```
xds_voice_dtmf_disconnect((void *)&cparm);
```

this will disconnect a voice channel DTMF detector from a user-specified input bus, stream, and timeslot with the parameters set in the *cparm* data structure

XDS Voice Resource Board
User Library Functions

This page was intentionally left blank.

Overview of XDS Voice Resource Board User Library

To ease the use of dealing with the board level (DSP) commands, Amtelco has created a user-interface function library. This library is intended to assist the user in developing real world applications written in the 'C' programming language. The majority of the command will take parameters such as a XDS Voice Resource board number (ie: 16) and a Voice Resource channel number on the board (ie: 0). Some functions will also use a data structure to set command parameters. These functions, parameters, and data structures are described in the following pages.

Please check your XDS board for voice resource compatibility before using these functions.

Return codes (errors) are defined in the xdslibvr.h and xdsioctl.h header files.

Additional notes

Once the initial playback / record gain and other parameters are set, playback and record will ignore any new parameters that will be written to the parameter table during playback / record. To adjust the playback gain during playback, use the xds_vr_set_play_gain() user function. To adjust the record gain during recording, use the xds_vr_set_record_gain().

Enabling or disabling full duplex will not work during playback/record. This should be enabled or disabled before playback/record.

Driver-to-application messages

These are messages that the driver uses to inform the library of events that the library needs to act on. They are passed in the **Msg** character array of the VOICE_RESOURCE_DATA data structure. The library will handle all of these messages for the user when the application is using the XDS library. These are NOT generated from the board itself.

They are as follows:

“Px” messages are requests for more voice data from the application to the driver, where **x** is the buffer number to fill.
A message of “P0” would indicate that the driver is requesting the application to fill user buffer 0 with more voice data.
A message of “P1” would indicate that the driver is requesting the application to fill user buffer 1 with more voice data.

“Rx” messages are updates to the application that another buffer has been filled with record voice data from the driver, where **x** is the buffer number that is filled.
A message of “R0” would indicate that the driver has filled the user buffer 0 with record voice data.
A message of “R1” would indicate that the driver has filled the user buffer 1 with record voice data.

“Sx” messages are playback stop messages from the driver to the application, where **x** is the reason for playback stop.
A message of “S0” would indicate that playback on a voice resource channel was stopped by the user (application).
A message of “S1” would indicate that playback on a voice resource channel was stopped via DTMF.
A message of “S2” would indicate that playback on a voice resource channel was stopped via end of file.

“Txr” messages are record done messages from the driver to the application, where **x** is the user buffer currently used and **r** is the reason for recording stopped.
A message of “T04” would indicate that recording has stopped in the driver because it has reached the limit of the user-assigned data buffer. In this case, **x** will always be ‘0’.
A message of “T05” would indicate that recording has stopped by the user (application) while filling the first user data buffer.
A message of “T15” would indicate that recording has stopped by the user (application) while filling the second user data buffer (if two buffers are used, ie record to file).
A message of “T06” would indicate that recording has via DTMF while filling the first user data buffer.
A message of “T16” would indicate that recording has stopped via DTMF while filling the second user data buffer (if two buffers are used, ie record to file).

When the recording is stopped, the driver tells the library the data length (in bytes) of the current user buffer filled in the **data_length1** variable of the VOICE_RESOURCE_DATA data structure. This is also passed to the user application in the **data_length** variable of the callback data structure.

“Dtx” messages are DTMF tones that are detected from the driver, where **t** is the DTMF tone detected, and the **x** is a ‘1’ when the tone starts playing and a ‘0’ when the tone finished playing. Valid DTMF tones are 0123456789ABCD*#.

A message of “D#1” would indicate the start of digit ‘#’.
A message of “D#0” would indicate the end of digit ‘#’.
A message of “D01” would indicate the start of digit ‘0’.
A message of “D00” would indicate the end of digit ‘0’.

xds_vr_enable_duplex

xds_vr_enable_duplex(board_number, voice_channel, mode);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used
unsigned char mode;	a character indicating to enable or disable full-duplex mode

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to enable or disable full-duplex mode on a voice resource channel.

Command Sent
0x00ccbb03, where **bb** is the voice resource channel and **cc** is the duplex mode to be set.

Returns
This function will return a XDS_SUCCESS (0) if successful or if the board is already in the mode requested.

A non-zero return value indicates an error condition:

XDS_CREATE_MUTEX_ERROR – could not create library mutex
XDS_CREATE_EVENT_ERROR – could not create library's signaling event mechanism
XDS_CREATE_RCV_THREAD_ERROR – could not create a receive thread in library

XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than an XDS Voice Resource board was selected
XDS_SYSTEM_ERROR	– system memory for the board parameters was unable to be allocated
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
XDS_PORT_BUSY	– the voice resource channel is currently in use
IOCTL()	- an IOCTL was called and returned a value from the IOCTL call

Comments
By default, each voice resource channel is configured as half-duplex. A value of 0x01 (ENABLE) for the third byte in the command will enable full-duplex mode and a value of 0x00 (DISABLE) for the third byte in the command will disable full-duplex mode on a given voice resource channel number.

The duplex mode may not be changed while the requested voice resource channel is currently playing or recording. This function will return an error indicating this condition.

Examples
xds_vr_enable_duplex(16, 0x00, ENABLE);
This will enable full-duplex mode on voice resource channel 0 of board 16
xds_vr_enable_duplex(16, 0x00, DISABLE);
This will disable full-duplex mode on voice resource channel 0 of board 16

xds_vr_enable_dtmf_detect

xds_vr_enable_dtmf_detect(board_number, voice_channel, mode);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used
unsigned char mode;	a character indicating to enable or disable full-duplex mode

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to enable or disable DTMF detection on a voice resource channel

Command Sent
0x00ccbb06, where **bb** is the voice resource channel and **cc** is the detection mode to be set.

Returns
This function will return a XDS_SUCCESS (0) if successful or if the board is already in the mode requested.

A non-zero return value indicates an error condition:

XDS_CREATE_MUTEX_ERROR – could not create library mutex
XDS_CREATE_EVENT_ERROR – could not create library's signaling event mechanism
XDS_CREATE_RCV_THREAD_ERROR – could not create a receive thread in library

XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than an XDS Voice Resource board was selected
XDS_SYSTEM_ERROR	– system memory for the board parameters was unable to be allocated
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL()	- an IOCTL was called and returned a value from the IOCTL call

Comments
By default, each voice resource channel has DTMF detection disabled. A value of 0x01 (ENABLE) for the third byte in the command will enable DTMF detection and a value of 0x00 (ENABLE) for the third byte in the command will disable DTMF detection on a given voice resource channel number.

The digit(s) detected by the board’s voice resource channel are reported to the user application in the callback data structure’s dtmf_tone variable.

Examples
xds_vr_enable_dtmf_detection(16, 0x00, ENABLE);
This will enable DTMF detection on voice resource channel 0 of board 16
xds_vr_enable_dtmf_detection(16, 0x00, DISABLE);
This will disable DTMF detection on voice resource channel 0 of board 16

xds_vr_dial_dtmf_string

xds_vr_dial_dtmf_string(board_number, voice_channel, dtmf_digits, &Callback_function);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used
char *dtmf_digits;	a character string of DTMF digits to dial
int Callback_function;	a user application callback function

typedef struct CallbackParm {

unsigned char	board_number;	board number that the callback is for
unsigned char	voice_channel;	number of the voice resource channel that generated the callback.
int	callback_code;	callback reports a callback code (reason).
unsigned char	dtmf_tone;	DTMF tone received by the board.
unsigned long	data_length;	the number of bytes for record buffer reported

} CallbackParms, *pCallbackParms;

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to dial a DTMF digit string on a voice resource channel.

Command Sent
0x0000bb0A, where **bb** is the voice resource channel number.

Returns
This function will return a XDS_SUCCESS (0) if successful or a non-zero return value indicates an error condition:

XDS_CREATE_MUTEX_ERROR – could not create library mutex
XDS_CREATE_EVENT_ERROR – could not create library's signaling event mechanism
XDS_CREATE_RCV_THREAD_ERROR – could not create a receive thread in library

XDS_WRONG_PARAMETER	– length of DTMF string exceeded limit (63 digits)
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than an XDS Voice Resource board was selected
XDS_SYSTEM_ERROR	– system memory for the board parameters was unable to be allocated
XDS_PORT_BUSY	– the voice resource channel is currently in use
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL()	- an IOCTL was called and returned a value from the IOCTL call

Comments
There is a limit of 63 digits that can be sent. If the voice resource channel is currently in use, this function will return an error to the user.

DTMF tones that can be sent:
‘**0**’, ‘**1**’, ‘**2**’, ‘**3**’, ‘**4**’, ‘**5**’, ‘**6**’, ‘**7**’, ‘**8**’, ‘**9**’, ‘*****’, ‘**#**’, ‘**A**’, ‘**B**’, ‘**C**’, ‘**D**’, ‘**U**’ – upper tone (941 Hz), ‘**L**’ – lower tone (697 Hz), ‘**X**’ – short pause, ‘**P**’ – long pause, ‘**N**’ – North American dial tone, ‘**E**’ – European dial tone, and ‘**a**’ <u>through</u> ‘**i**’ – special and programmable tones.

Example

xds_vr_dial_dtmf_string(16, 0x00, “0123456789ABCD*#”, &Callback_function);
This will dial 0123456789ABCD*# on voice resource channel 0 of board 16, then call the Callback_function() callback function in the user application when finished dialing.

xds_vr_generate_custom_tone

xds_vr_generate_custom_tone(board_number, voice_channel, &VrUserTone, &Callback_function);

```
unsigned char board_number;        a value indicating which board the command is for
unsigned char voice_channel;       a character indicating the voice resource channel to be used

typedef struct VRUSERTONE{
unsigned short    first_tone_freq;        first tone frequency (in Hz)
unsigned short    first_output_level;     output level (in –dBm) of the first tone
unsigned short    second_tone_freq;       second tone frequency (in Hz)
unsigned short    second_output_level;    output level (in –dBm) of the second tone
unsigned short    on_time;                on time for the tone (in 50mS increments)
unsigned short    off_time;               off time for the tone (in 50mS increments)
unsigned short    number_repetiton;       number of repetitions for the tone
}VrUserTone, *pVrUserTone;

int Callback_function;             a user application callback function

typedef struct CallbackParm {
unsigned char  board_number;       board number that the callback is for
unsigned char  voice_channel;      number of the voice resource channel that generated
                                   the callback
int            callback_code;      callback reports a callback code (reason)
unsigned char  dtmf_tone;          DTMF tone received by the board
unsigned long  data_length;        number of bytes for record buffer reported
} CallbackParms, *pCallbackParms;
```

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to generate a custom user tone on a voice resource channel.

Command Sent
0x0000bb0B, where **bb** is the voice resource channel number.

Returns
This function will return a XDS_SUCCESS (0) if successful or a non-zero return value indicates an error condition:

XDS_CREATE_MUTEX_ERROR – could not create library mutex
XDS_CREATE_EVENT_ERROR – could not create library's signaling event mechanism
XDS_CREATE_RCV_THREAD_ERROR – could not create a receive thread in library

XDS_WRONG_PARAMETER	– one of the custom tone parameters was invalid
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than an XDS Voice Resource board was selected
XDS_SYSTEM_ERROR	– system memory for the board parameters was unable to be allocated
XDS_PORT_BUSY	– the voice resource channel is currently in use
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL()	- an IOCTL was called and returned a value from the IOCTL call

Comments
The first tone frequency and second tone frequency ranges are from 0 to 3999 Hz, if a single tone is to be generated, the second tone frequency should be set to 0. A "Bong" tone can be defined by negating the frequency. However, "Bong" tones will only work correctly for a single repetition.

The output level of the first and second frequency is in –dBm and ranges from 0 to 63. A value of 63 (0x003F) or greater will disable the tone.

The on and off time for each tone is in 50 mS increments. The range is 0 to 655 (0x0000 to 0x028F)

The number of repetitions is the number of times the tone will be sounded and can range from 0 - 0x7fff.

Example

```
VrUserTone.first_tone_freq = 400;
VrUserTone.first_output_level = 10;
VrUserTone.second_tone_freq = 2440;
VrUserTone.second_output_level = 20;
VrUserTone.on_time = 20;
VrUserTone.off_time = 20;
VrUserTone.number_repetiton = 1;

xds_vr_generate_custom_tone(16, 0, &VrUserTone, &Callback_function);
```

This will generate a tone with the first frequency of 400 Hz @ -10dBm and a second frequency of 1440 Hz @ -20dBm. The on time and off time for this tone will be 1 second and will sound once on voice resource channel 0 of board 16.

When this tone is finished sounding, the Callback_function() callback function in the user application will then be called.

xds_vr_stop_play

xds_vr_stop_play(board_number, voice_channel);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to stop playback on a voice resource channel.

Command Sent
0x0000bb04, where **bb** is the voice resource channel.

Returns
This function will return a XDS_SUCCESS (0) if successful or if the voice resource channel is currently not in use.

A non-zero return value indicates an error condition:

XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than an XDS Voice Resource board was selected
XDS_SYSTEM_ERROR	– system memory for the board parameters was unable to be allocated
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL()	- an IOCTL was called and returned a value from the IOCTL call

Comments
This function is used to stop voice playback manually from the application.

Examples

xds_vr_stop_play(16, 0x00);
This will stop playback on voice resource channel 0 of board 16

xds_vr_stop_play(16, 0x01);
This will stop playback on voice resource channel 1 of board 16

xds_vr_stop_record

xds_vr_stop_record(board_number, voice_channel);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to stop recording on a voice resource channel.

Command Sent
0x0000bb05, where **bb** is the voice resource channel.

Returns
This function will return a XDS_SUCCESS (0) if successful or if the voice resource channel is currently not in use.

A non-zero return value indicates an error condition:

XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than an XDS Voice Resource board was selected
XDS_SYSTEM_ERROR	– system memory for the board parameters was unable to be allocated
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL()	- an IOCTL was called and returned a value from the IOCTL call

Comments
This function is used to stop voice recording manually from the application.

Examples

xds_vr_stop_record(16, 0x00);
This will stop recording on voice resource channel 0 of board 16

xds_vr_stop_record(16, 0x01);
This will stop recording on voice resource channel 1 of board 16

xds_vr_play_file

xds_vr_play_file(board_number, voice_channel, &PlayFileParm, &Callback_function);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used

```
typedef struct PLAYFILEPARM{
short           times_to_play;      number of time to play file
short           data_format;        the data format of the file to played
unsigned char   file_type;          the file type of the file to be played
unsigned char   *FileName;          filename of the file to be played
} PlayFileParm, *pPlayFileParm;

int Callback_function;              a user application callback function

typedef struct CallbackParm {
unsigned char   board_number;       board number that the callback is for
unsigned char   voice_channel;      number of the voice resource channel that generated
                                    the callback
int             callback_code;      callback reports a callback code (reason)
unsigned char   dtmf_tone;          DTMF tone received by the board
unsigned long   data_length;        number of bytes for record buffer reported
} CallbackParms, *pCallbackParms;
```

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to play a voice file on a given voice resource channel.

Command Sent
0x00ccbb01, where **bb** is the voice resource channel and **cc** is the data format of the media to be played.

Returns
This function will return a XDS_SUCCESS (0) if successful. A non-zero return value indicates an error condition:

XDS_CREATE_MUTEX_ERROR	**–** could not create library mutex
XDS_CREATE_EVENT_ERROR	**–** could not create library’s signaling event mechanism
XDS_CREATE_RCV_THREAD_ERROR	**–** could not create a receive thread in library
XDS_WRONG_PARAMETER	– invalid voice file parameters were used
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than a voice resource board was selected
XDS_PORT_BUSY	– the voice resource channel is currently playing back
XDS_SYSTEM_ERROR	– system memory for user playback buffer was unable to be allocated
XDS_FILE_NOTFOUND	**-** voice file name is blank or the file is not present
XDS_BAD_FILE	**–** voice file other than supported formats was chosen for play
ILL_ARG	**–** a value of less then 1 was passed for times_to_play
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL	- an IOCTL was called and returns a value from the IOCTL call

Comments
If the user attempts to playback on a voice resource channel that is in use, it will result in an error. Playback must be stopped first before playing a different file.

If the user enables DTMF detection on a voice resource channel for playback, they will need to call the xds_vr_stop_play() library function before using the same voice resource channel again. This is done so that if the user wants to still detect digits, they can even though playback is done. Not calling the stop function may result in a resource error returned the next time the user tries to use the voice resource channel.

When writing an application the user will want to watch the callback mechanism (form the XDS voice resource library) for any 'read disk' errors during playback. There is an example of this in the xds_vr_play_file project included in the driver package. If the user does receive a callback error of XDS_READ_FROM_DISK_FAILED_DURING_PLAYBACK, a hard disk access error more than likely has occurred.

The options for **times_to_play** are:
0 – 32767, or XDS_REPEAT_INFINITY (infinity (-1))
This is the number of times to playback a file.

The options for **data_format** are:

XDS_DATA_16BIT_LINEAR	- 16 bit linear PCM file
XDS_DATA_ALAW	- 8 bit CCITT A-Law file
XDS_DATA_ULAW	- 8 bit CCITT u-Law file
XDS_DATA_ADPCM_6K	- 6 kHz ADPCM file
XDS_DATA_ADPCM_8K	- 8 kHz ADPCM file

This is the data format of the file to be played back.

The options for **file_type** are:

XDS_WAVE_FILE	- a standard wav file (with header – A-Law, u-Law, and 16 bit linear)
XDS_RAW_FILE	- a raw file (no header – 6 kHz and 8 kHz ADPCM)

This is the file type of the file to be played back.

Filename is a pointer to an ASCII string representing the file to be played back.

Examples

```
PlayFileParm.times_to_play = 1;
PlayFileParm.FileName = (unsigned char *) “one_u.wav”;

xds_vr_play_file(17, 5, &PlayFileParm, &Callback_function);
```

This will open and play 8 bit u-Law .wav file “one_u.wav” once on voice resource channel 5 of board 17. Note that the file type and format for wave files are determined by the XDS Voice Resource library and are not needed in the application.

When the file finishes playing (from end of file being reached), DTMF termination, or user stops the playback manually (from the user application), the user application callback function Callback_function() will be called.

```
PlayFileParm.times_to_play = 2;
PlayFileParm.data_format = XDS_DATA_ADPCM_8K;
PlayFileParm.file_type = XDS_RAW_FILE;
PlayFileParm.FileName = (unsigned char *) “adpcm.8k”;

xds_vr_play_file(17, 31, &PlayFileParm, &Callback_function);
```

This will open and play a raw (data only) 8 kHz ADPCM file named “adpcm.8k” two times on voice resource channel 31 of board 17.

When the file finishes playing (from end of file) or the user stops the playback manually (from the application), the user application callback function Callback_function() will be called.

xds_vr_play_file_index

xds_vr_play_file_index(board_number, voice_channel, &PlayFileIndexParm,
&Callback_function);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used

typedef struct PLAYFILEINDEXPARM{

short	times_to_play;	number of times to playback the file list (indexed)
short	data_format;	the data format of the files to played
unsigned char	file_type;	the file type of the files to be played
char	**pFiles;	pointer to a list of file names to be played

} PlayFileIndexParm, *pPlayFileIndexParm;

int Callback_function;	a user application callback function

typedef struct CallbackParm {

unsigned char	board_number;	board number that the callback is for
unsigned char	voice_channel;	number of the voice resource channel that generated the callback
int	callback_code;	callback reports a callback code (reason)
unsigned char	dtmf_tone;	DTMF tone received by the board
unsigned long	data_length;	number of bytes for record buffer reported

} CallbackParms, *pCallbackParms;

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to play an index of voice files (of the same file type and data format) on a given voice resource channel.

Command Sent
0x00ccbb01, where **bb** is the voice resource channel and **cc** is the data format of the media to be played.

Returns
This function will return a XDS_SUCCESS (0) if successful. A non-zero return value indicates an error condition:

XDS_CREATE_MUTEX_ERROR	**–** could not create library mutex
XDS_CREATE_EVENT_ERROR	**–** could not create library's signaling event mechanism
XDS_CREATE_RCV_THREAD_ERROR	**–** could not create a receive thread in library
XDS_WRONG_PARAMETER	– invalid voice file parameters were used
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than a voice resource board was selected
XDS_PORT_BUSY	– the voice resource channel is currently playing back
XDS_SYSTEM_ERROR	– system memory for user playback buffer was unable to be allocated
XDS_FILE_NOTFOUND	**-** voice file name is blank or the file is not present
XDS_BAD_FILE	**–** voice file other than supported formats was chosen for play
XDS_FILES_NOT_TERMINATED	**-** all of the files in the index are not the same type
ILL_ARG	**–** a value of less then 1 was passed for times_to_play
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL	- an IOCTL was called and returns a value from the IOCTL call

Comments
If the user attempts to playback on a voice resource channel that is in use, it will result in an error. Playback must be stopped first before playing a different file.

If the user enables DTMF detection on a voice resource channel for playback, they will need to call the xds_vr_stop_play() library function before using the same voice resource channel again. This is done so that if the user wants to still detect digits, they can even though playback is done. Not calling the stop function may result in a resource error returned the next time the user tries to use the voice resource channel.

All of the files in the indexed play list must be of the same and defined file type and data format.

When writing an application the user will want to watch the callback mechanism (form the XDS voice resource library) for any 'read disk' errors during playback. There is an example of this in the xds_vr_play_file_index project included in the driver package. If the user does receive a callback error of XDS_READ_FROM_DISK_FAILED_DURING_PLAYBACK, a hard disk access error more than likely has occurred.

The options for **times_to_play** are:
1 – 32767, or XDS_REPEAT_INFINITY (infinity (-1))
This is the number of times to playback the file list (indexed).

The options for **data_format** are:

XDS_DATA_16BIT_LINEAR	- 16 bit linear PCM file
XDS_DATA_ALAW	- 8 bit CCITT A-Law file
XDS_DATA_ULAW	- 8 bit CCITT u-Law file
XDS_DATA_ADPCM_6K	- 6 kHz ADPCM file
XDS_DATA_ADPCM_8K	- 8 kHz ADPCM file

This is the data format of the file to be played back.

The options for **file_type** are:

XDS_WAVE_FILE	- a standard wav file (with header – A-Law, u-Law, and 16 bit linear)
XDS_RAW_FILE	- a raw file (no header – 6 kHz and 8 kHz ADPCM)

This is the file type of the file to be played back.

pFiles is a pointer to a list of files to be played back.

Examples

```
PlayFileIndexParm.times_to_play = 1;

strcpy(file_list[0], “one_u.wav”);
strcpy(file_list[1], “two_u.wav”);
file_list[2] = NULL;

PlayFileIndexParm.pFiles = file_list;

xds_vr_play_file_index(17, 5, &PlayFileIndexParm, &Callback_function);
```

This will open and play 8 bit u-Law .wav files “one_u.wav” and “two_u.wav” once on voice resource channel 5 of board 17. Note that the file type and format for wave files are determined by the XDS Voice Resource library and are not needed in the application.

When the files finish playing (from end of files), DTMF termination, or user stops the playback manually (from the application) the user application callback function Callback_function() will be called.

```
PlayFileIndexParm.times_to_play = 2;
PlayFileIndexParm.data_format = XDS_DATA_ADPCM_8K;
PlayFileIndexParm.file_type = XDS_RAW_FILE;

strcpy(file_list[0], “adpcm.8k”);
strcpy(file_list[1], “test.8k”);
file_list[2] = NULL;

PlayFileIndexParm.pFiles = file_list;

xds_vr_play_file_index(17, 31, &PlayFileIndexParm, &Callback_function);
```

This will open and play raw (data only) 8 kHz ADPCM files “adpcm.8k” and “test.8k” twice on voice resource channel 31 of board 17 while DTMF.

When the files finish playing (from end of files) or user manually (from the application) stops the playback the user application callback function Callback_function() will be called.

xds_vr_play_buffer

xds_vr_play_buffer(board_number, voice_channel, &PlayBufParm, &Callback_function);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used

```
typedef struct PLAYBUFPARM{
short           times_to_play;      number of time to playback the voice buffer
short           data_format;        the data format of the file to played
unsigned char   file_type;          the file type of the file to be played
unsigned long   data_length;        length of data in data buffer (in dwaords)
void            *pBuffer;           pointer to a data buffer to be played
} PlayBufParm, *pPlayBufParm;

int Callback_function;              a user application callback function

typedef struct CallbackParm {
unsigned char   board_number;       board number that the callback is for
unsigned char   voice_channel;      number of the voice resource channel that generated
                                    the callback
int             callback_code;      callback reports a callback code (reason)
unsigned char   dtmf_tone;          DTMF tone received by the board
unsigned long   data_length;        number of bytes for record buffer reported
} CallbackParms, *pCallbackParms;
```

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to play a buffered (in user memory space) voice file on a given voice resource channel.

Command Sent
0x00ccbb01, where **bb** is the voice resource channel and **cc** is the data format of the media to be played.

Returns
This function will return a XDS_SUCCESS (0) if successful. A non-zero return value indicates an error condition:

XDS_CREATE_MUTEX_ERROR –	could not create library mutex
XDS_CREATE_EVENT_ERROR –	could not create library's signaling event mechanism
XDS_CREATE_RCV_THREAD_ERROR –	could not create a receive thread in library
XDS_WRONG_PARAMETER	– invalid voice file parameters were used
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than a voice resource board was selected
XDS_PORT_BUSY	– the voice resource channel is currently playing back
XDS_BAD_FILE	**–** voice file other than supported formats was chosen for play
ILL_ARG	**–** a value of less then 1 was passed for times_to_play
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL	- an IOCTL was called and returns a value from the IOCTL call

Comments
If the user attempts to playback on a voice resource channel that is in use, it will result in an error. Playback must be stopped first before playing a different file.

If the user enables DTMF detection on a voice resource channel for playback, they will need to call the xds_vr_stop_play() library function before using the same voice resource channel again. This is done so that if the user wants to still detect digits, they can even though playback is done. Not calling the stop function may result in a resource error returned the next time the user tries to use the voice resource channel.

When writing an application the user will want to watch the callback mechanism (form the XDS voice resource library) for any 'read disk' errors during playback. There is an example of this in the xds_vr_play_buffer project included in the driver package. If the user does receive a callback error of XDS_READ_FROM_DISK_FAILED_DURING_PLAYBACK, a hard disk access error more than likely has occurred.

The options for **times_to_play** are:
0 – 32767, or XDS_REPEAT_INFINITY (infinity (-1))
This is the number of times to playback the voice buffer.

The options for **data_format** are:

XDS_DATA_16BIT_LINEAR	- 16 bit linear PCM file
XDS_DATA_ALAW	- 8 bit CCITT A-Law file
XDS_DATA_ULAW	- 8 bit CCITT u-Law file
XDS_DATA_ADPCM_6K	- 6 kHz ADPCM file
XDS_DATA_ADPCM_8K	- 8 kHz ADPCM file

This is the data format of the file to be played back.

The options for **file_type** are:

XDS_WAVE_FILE	- a standard wav file (with header – A-Law, u-Law, and 16 bit linear)
XDS_RAW_FILE	- a raw file (no header – 6 kHz and 8 kHz ADPCM)

This is the file type of the file to be played back.

data_length is the size (in bytes) of the user buffer to be played back.

pBuffer is a pointer to a user buffer to be played back.

Examples

```
PlayBufParm.times_to_play = 1;
PlayBufParm.data_format = XDS_DATA_ULAW;
PlayBufParm.file_type = XDS_WAVE_FILE;
PlayBufParm.data_length = sizeof(pUserBuffer);
PlayBufParm.pBuffer = pUserBuffer;

xds_vr_play_buffer(17, 5, &PlayBufParm, &Callback_function);
```

This will play user buffer *pUserbuffer* once on voice resource channel 5 of board 17 while DTMF digit detection is enabled.

When the buffer finishes playing (from end of buffer being reached), DTMF termination, or user stops the playback manually (from the user application), the user application callback function Callback_function() will be called.

```
PlayBufParm.times_to_play = 1;
PlayBufParm.data_format = XDS_DATA_ALAW;
PlayBufParm.file_type = XDS_WAVE_FILE;
PlayBufParm.data_length = sizeof(pUserBuffer1);
PlayBufParm.pBuffer = pUserBuffer1;

xds_vr_play_buffer(17, 6, &PlayBufParm, &Callback_function);
```

This will play user buffer *pUserbuffer1* once on voice resource channel 5 of board 17.

When the buffer finishes playing (from end of buffer being reached) or the user stops the playback manually (from the application), the user application callback function Callback_function() will be called.

xds_vr_play_buffer_index

xds_vr_play_buffer_index(board_number, voice_channel, &PlayBufIndexParm,
&Callback_function);

```
unsigned char board_number;        a value indicating which board the command is for
unsigned char voice_channel;       a character indicating the voice resource channel to be used

typedef struct PLAYBUFINDEXPARM{
short           times_to_play;     number of time to playback the buffer list (indexed)
short           data_format;       the data format of the file to played
unsigned char   file_type;         the file type of the file to be played
BufferInfo      **pBuffers;        a pointer to a data structure with the buffer information
} PlayBufIndexParm, *pPlayBufIndexParm;

typedef struct BUFFERINFO{
unsigned char *pBuffer;            a pointer to a buffer
unsigned long  data_length;        length of buffer
}BufferInfo, *pBufferInfo;

int Callback_function;             a user application callback function

typedef struct CallbackParm {
unsigned char   board_number;      board number that the callback is for
unsigned char   voice_channel;     number of the voice resource channel that generated
                                   the callback
int             callback_code;     callback reports a callback code (reason)
unsigned char   dtmf_tone;         DTMF tone received by the board
unsigned long   data_length;       number of bytes for record buffer reported
} CallbackParms, *pCallbackParms;
```

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to play a list of buffered (in user memory space) voice files (of the same file type and data format) on a given voice resource channel.

Command Sent
0x00ccbb01, where **bb** is the voice resource channel and **cc** is the data format of the media to be played.

Returns
This function will return a XDS_SUCCESS (0) if successful. A non-zero return value indicates an error condition:

XDS_CREATE_MUTEX_ERROR – could not create library mutex
XDS_CREATE_EVENT_ERROR – could not create library's signaling event mechanism
XDS_CREATE_RCV_THREAD_ERROR – could not create a receive thread in library

XDS_WRONG_PARAMETER	– invalid voice file parameters were used
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than a voice resource board was selected
XDS_FILES_NOT_TERMINATED	**-** all of the buffers in the index are not the same type
XDS_PORT_BUSY	– the voice resource channel is currently playing back
XDS_BAD_FILE	**–** voice file other than supported formats was chosen for play
ILL_ARG	**–** a value of less then 1 was passed for times_to_play
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL	- an IOCTL was called and returns a value from the IOCTL call

Comments
If the user attempts to playback on a voice resource channel that is in use, it will result in an error. Playback must be stopped first before playing a different file.

If the user enables DTMF detection on a voice resource channel for playback, they will need to call the xds_vr_stop_play() library function before using the same voice resource channel again. This is done so that if the user wants to still detect digits, they can even though playback is done. Not calling the stop function may result in a resource error returned the next time the user tries to use the voice resource channel.

All of the buffers in the indexed play list must be of the same and defined file type and data format.

The options for **times_to_play** are:
0 – 32767, or XDS_REPEAT_INFINITY (infinity (-1))
This is the number of times to playback the buffer list (indexed).

The options for **data_format** are:

XDS_DATA_16BIT_LINEAR	- 16 bit linear PCM file
XDS_DATA_ALAW	- 8 bit CCITT A-Law file
XDS_DATA_ULAW	- 8 bit CCITT u-Law file
XDS_DATA_ADPCM_6K	- 6 kHz ADPCM file
XDS_DATA_ADPCM_8K	- 8 kHz ADPCM file

This is the data format of the file to be played back.

The options for **file_type** are:

XDS_WAVE_FILE	- a standard wav file (with header – A-Law, u-Law, and 16 bit linear)
XDS_RAW_FILE	- a raw file (no header – 6 kHz and 8 kHz ADPCM)

This is the file type of the file to be played back.

pBuffers is a list of buffers to be played back.

BufferInfo.pBuffer is a pointer to a buffer to be played back.
BufferInfo.pBuffers is the size of the buffer to be played back.

Examples

```
BufferInfo[0].pBuffer = pUserBuffer0;
BufferInfo[0].data_length = sizeof(pUserBuffer0);

BufferInfo[1].pBuffer = pUserBuffer1;
BufferInfo[1].data_length = sizeof(pUserBuffer1);

BufferInfo[2].pBuffer = NULL;

PlayBufIndexParm.times_to_play = 1;
PlayBufIndexParm.data_format = XDS_DATA_ULAW;
PlayBufIndexParm.file_type = XDS_WAVE_FILE;
PlayBufIndexParm.pBuffer = BufferInfo;

xds_vr_play_buffer_index(17, 5, &PlayBufIndexParm, &Callback_function);
```

This will play the user buffers (indexed) *pUserbuffer0* and *pUserbuffer1* once on voice resource channel 5 of board 17. When the buffers finish playing (from end of buffers being reached), DTMF termination, or user stops the playback manually (from the user application), the user application callback function Callback_function() will be called.

```
BufferInfo[0].pBuffer = pUserBuffer2;
BufferInfo[0].data_length = sizeof(pUserBuffer2);

BufferInfo[1].pBuffer = pUserBuffer3;
BufferInfo[1].data_length = sizeof(pUserBuffer3);

BufferInfo[2].pBuffer = NULL;

PlayBufIndexParm.times_to_play = 2;
PlayBufIndexParm.data_format = XDS_DATA_ALAW;
PlayBufIndexParm.file_type = XDS_WAVE_FILE;
PlayBufIndexParm.pBuffer = BufferInfo;

xds_vr_play_buffer_index(17, 30, &PlayBufIndexParm, &Callback_function);
```

This will play the user buffers (indexed) *pUserbuffer2* and *pUserbuffer3* twice on voice resource channel 30 of board 17. When the buffers finish playing (from end of buffers being reached) or the user stops the playback manually (from the application), the user application callback function Callback_function() will be called.

xds_vr_set_play_gain

xds_vr_set_play_gain(board_number, voice_channel, gain, gain_type);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used
unsigned char gain;	a character indicating the amount of gain to be used
unsigned char gain_type;	a character indicating if the gain is relative or absolute to the current gain setting

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to control the gain on a voice resource channel during playback.

Command Sent
0xddccbb07, where **bb** is the voice resource channel, **cc** is the gain value, and **dd** is the gain setting (relative or absolute).

Returns
This function will return a XDS_SUCCESS (0) if successful or a non-zero return value indicates an error condition:

XDS_WRONG_PARAMETER	– an invalid value other than relative or absolute was chosen for the gain_type
XDS_WRONG_BOARD	– a board other than an XDS Voice Resource board was selected
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL()	- an IOCTL was called and returned a value from the IOCTL call

Comments
This function is used to adjust the playback gain of a voice resource channel during playback. The range of gain is from (-64) dB to 20 dB. Gain can be set to an absolute level or a level relative to the current gain setting.

Examples

xds_vr_set_play_gain(16, 0x00, 1, RELATIVE_GAIN);
This will increase the gain on channel 0 by 1 dB relative to the current gain setting on board 16.

xds_vr_set_play_gain(16, 0x01, 2, ABSOLUTE_GAIN);
This will increase the gain on channel 1 by 2 dB absolutely on board 16.

xds_vr_record_file

xds_vr_record_file(board_number, voice_channel, &RecordFileParm,
&Callback_function);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used

typedef struct RECODRFILEPARM{

short	data_format;	the data format of the files to recorded
unsigned char	file_type;	the file type of the files to be recorded
unsigned char	silence_suppression;	option to suppress silence in recording
unsigned char	*FileName;	filename of the file to be recorded

} RecordFileParm, *pRecordFileParm;

int Callback_function;	a user application callback function

typedef struct CallbackParm {

unsigned char	board_number;	board number that the callback is for
unsigned char	voice_channel;	number of the voice resource channel that generated the callback
int	callback_code;	callback reports a callback code (reason)
unsigned char	dtmf_tone;	DTMF tone received by the board
unsigned long	data_length;	number of bytes for record buffer reported

} CallbackParms, *pCallbackParms;

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to record a voice file on a given voice resource channel.

Command Sent
0x00ccbb02, where **bb** is the voice resource channel and **cc** is the data format of the media to be recorded.

Returns
This function will return a XDS_SUCCESS (0) if successful. A non-zero return value indicates an error condition:

ILL_ARG	**-** an invalid silence suppression mode chosen
XDS_CREATE_MUTEX_ERROR	**–** could not create library mutex
XDS_CREATE_EVENT_ERROR	**–** could not create library's signaling event mechanism
XDS_CREATE_RCV_THREAD_ERROR	**–** could not create a receive thread in library
XDS_WRONG_PARAMETER	– invalid voice file parameters were used
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than a voice resource board was selected
XDS_PORT_BUSY	– the voice resource channel is currently recording
XDS_SYSTEM_ERROR	– system memory for user record buffer was unable to be allocated
XDS_FILE_NOTFOUND	- voice file name is blank or the file is not present
XDS_WRITE_TO_DISK_FAILED	- header for voice file (to record) failed to write to disk
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL	- an IOCTL was called and returns a value from the IOCTL call

Comments
If the user attempts to record on a voice resource channel that is in use, it will result in an error. Recording must be stopped first before recording a new file.

If the user enables DTMF detection on a voice resource channel for recording, they will need to call the xds_vr_stop_record() library function before using the same voice resource channel again. This is done so that if the user wants to still detect digits, they can even though recording is done. Not calling the stop function may result in a resource error returned the next time the user tries to use the voice resource channel.

When writing an application the user will want to watch the callback mechanism (form the XDS voice resource library) for any 'write disk' errors during recording. There is an example of this in the xds_vr_record_file project included in the driver package. If the user does receive a callback error of XDS_WRITE_TO_DISK_FAILED_DURING_RECORD, a hard disk access error more than likely has occurred.

At the time of this release only 8 bit A-Law, 8 bit u-Law, 16 bit linear, and 8 kHz ADPCM are supported file types and data formats for recording.

The options for **data_format** are:

XDS_DATA_16BIT_LINEAR	- 16 bit linear PCM file
XDS_DATA_ALAW	- 8 bit CCITT A-Law file
XDS_DATA_ULAW	- 8 bit CCITT u-Law file
XDS_DATA_ADPCM_8K	- 8kHz ADPCM file

This is the data format of the file to be played back.

The options for **file_type** are:

XDS_WAVE_FILE	- a standard wav file (with header – A-Law, u-Law, and 16 bit linear)
XDS_RAW_FILE	- a raw file (no header – 6 kHz and 8 kHz ADPCM)

This is the file type of the file to be played back.

silence_suppression is either ENABLE to suppress (discard) silence detected during the recording, or DISABLE to not suppress (keep) the silence detected during recording.

Filename is a pointer to an ASCII string representing the file to be recorded.

Examples

```
RecordFileParm.data_format = XDS_DATA_ULAW;
RecordFileParm.file_type = XDS_WAVE_FILE;
RecordFileParm.silence_suppression = ENABLE;
RecordFileParm.FileName = (unsigned char *) “record_u.wav”;

xds_vr_record_file(17, 3, &RecordFileParm, &Callback_function);
```

This will record and save an 8 bit u-Law .wav file named “record_u.wav” from voice resource channel 3 of board 17. This will also suppress (discard) the silence detected during a recording.

When the file finishes recording from DTMF termination or user stops recording manually (from the user application), the user application callback function Callback_function() will be called.

```
RecordFileParm.data_format = XDS_DATA_ADPCM_8K;
RecordFileParm.file_type = XDS_RAW_FILE;
RecordFileParm.silence_suppression = DISABLE;
RecordFileParm.FileName = (unsigned char *) “record.8k”;

xds_vr_record_file(17, 31, &RecordFileParm, &Callback_function);
```

This will record and save a raw (data only) 8 kHz ADPCM file named “adpcm.8k” from voice resource channel 31 of board 17. This will not suppress (keep) the silence detected during a recording.

When the file finishes recording from user stopping the recording manually (from the user application) or the user application callback function Callback_function() will be called.

xds_vr_record_buffer

xds_vr_record_buffer(board_number, voice_channel, &RecordBufParm, &Callback_function);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used

typedef struct RECODRDBUFPARM{

short	data_format;	the data format of the files to recorded
unsigned long	buffer_length;	size of user buffer (in bytes)
unsigned char	silence_suppression;	option to suppress silence in recording
void	*pBuffer;	pointer to the user application record buffer

} RecordBufParm, *pRecordBufParm;

int Callback_function;	a user application callback function

typedef struct CallbackParm {

unsigned char	board_number;	board number that the callback is for
unsigned char	voice_channel;	number of the voice resource channel that generated the callback
int	callback_code;	callback reports a callback code (reason)
unsigned char	dtmf_tone;	DTMF tone received by the board
unsigned long	data_length;	number of bytes for record buffer reported

} CallbackParms, *pCallbackParms;

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to record a buffered (in user memory space) voice file on a given voice resource channel.

Command Sent
0x00ccbb02, where **bb** is the voice resource channel and **cc** is the data format of the media to be recorded.

Returns
This function will return a XDS_SUCCESS (0) if successful. A non-zero return value indicates an error condition:

ILL_ARG	- an invalid silence suppression mode chosen
XDS_CREATE_MUTEX_ERROR –	could not create library mutex
XDS_CREATE_EVENT_ERROR –	could not create library's signaling event mechanism
XDS_CREATE_RCV_THREAD_ERROR –	could not create a receive thread in library
XDS_WRONG_PARAMETER	– invalid voice file parameters were used
XDS_BOARD_NOT_ACTIVE	– board selected is not present
XDS_WRONG_BOARD	– a board other than a voice resource board was selected
XDS_PORT_BUSY	– the voice resource channel is currently recording
XDS_SYSTEM_ERROR	– system memory for user record buffer was unable to be allocated
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL	- an IOCTL was called and returns a value from the IOCTL call

Comments
If the user attempts to record on a voice resource channel that is in use, it will result in an error. Recording must be stopped first before recording a new file.

If the user enables DTMF detection on a voice resource channel for recording, they will need to call the xds_vr_stop_record() library function before using the same voice resource channel again. This is done so that if the user wants to still detect digits, they can even though recording is done. Not calling the stop function may result in a resource error returned the next time the user tries to use the voice resource channel.

At the time of this release only 8 bit A-Law, 8 bit u-Law, 16 bit linear, and 8 kHz ADPCM are supported file types and data formats for recording.

The options for **data_format** are:

XDS_DATA_16BIT_LINEAR	- 16 bit linear PCM file
XDS_DATA_ALAW	- 8 bit CCITT A-Law file
XDS_DATA_ULAW	- 8 bit CCITT u-Law file
XDS_DATA_ADPCM_8K	- 8kHz ADPCM file

This is the data format of the file to be played back.

buffer_length is the maximum size (in bytes) of the user data buffer to record to.

silence_suppression is either ENABLE to suppress (discard) silence detected during the recording, or DISABLE to not suppress (keep) the silence detected during recording.

pBuffer is a pointer to an user data buffer (unsigned long) to record to.

Examples

```
RecordBufParm.data_format = XDS_DATA_ULAW;
RecordBufParm.buffer_length = 65536;
RecordBufParm.silence_suppression = ENABLE;
RecordBufParm.pBuffer = pBuffer;

xds_vr_record_buffer(17, 3, &RecordBufParm, &Callback_function);
```

This will record and save an up to 64k of 8 bit u-Law buffered voice into application buffer *pBuffer* from voice resource channel 3 of board 17. This will suppress (discard) the silence detected during a recording.

When the buffer finishes recording from DTMF termination, user buffer being completely filled, or user stops recording manually (from the user application), the user application callback function Callback_function() will be called.

```
RecordBufParm.data_format = XDS_DATA_ADPCM_8K;
RecordBufParm.buffer_length = 65536;
RecordBufParm.silence_suppression = DISABLE;
RecordBufParm.pBuffer = pBuffer1;

xds_vr_record_buffer(17, 31, &RecordBufParm, &Callback_function);
```

This will record and save an up to 64k of 8 kHz ADPCM buffered voice into application buffer *pBuffer1* from voice resource channel 31 of board 17. This will not suppress (keep) the silence detected during a recording.

When the buffer finishes recording from the user buffer being completely filled or
user stops recording manually (from the user application), the user application callback function Callback_function() will be called.

xds_vr_set_record_gain

xds_vr_set_record_gain(board_number, voice_channel, gain, gain_type);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used
unsigned char gain;	a character indicating the amount of gain to be used
unsigned char gain_type;	a character indicating if the gain is relative or absolute to the current gain setting

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This function is used to control the gain on a voice resource channel during recording.

Command Sent
0xddccbb08, where **bb** is the voice resource channel, **cc** is the gain value, and **dd** is the gain setting (relative or absolute).

Returns
This function will return a XDS_SUCCESS (0) if successful or a non-zero return value indicates an error condition:

XDS_WRONG_PARAMETER	– an invalid value other than relative or absolute was chosen for the gain_type
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL()	- an IOCTL was called and returned a value from the IOCTL call

Comments
This function is used to adjust the record gain of a voice resource channel during recording. The range of gain is from (-64) dB to 20 dB. Gain can be set to an absolute level or a level relative to the current gain setting.

Examples

xds_vr_set_record_gain(16, 0x00, 1, RELATIVE_GAIN);
This will increase the gain on channel 0 by 1 dB relative to the current gain setting on board 16.

xds_vr_set_record_gain(16, 0x01, 2, ABSOLUTE_GAIN);
This will increase the gain on channel 1 by 2 dB absolutely on board 16.

xds_vr_set_port_parameters

xds_vr_set_port_parameters(board_number, voice_channel, &VrSetPortParameters);

unsigned char board_number;	a value indicating which board the command is for
unsigned char voice_channel;	a character indicating the voice resource channel to be used

typedef struct VRSETPORTPARAMETERS{

unsigned long	tag_value;	which parameters to be set in this function
unsigned short	play_dtmf_terminate_bitmap;	the DTMF tone bitmap to use for terminate play
unsigned short	record_dtmf_terminate_bitmap;	the DTMF tone bitmap to use for terminate record
unsigned char	play_gain;	amount of gain (in dB) to use for playback
unsigned char	record_gain;	amount of gain (in dB) to use for record
unsigned char	agc_enable;	AGC enable (record only) parameter
unsigned char	stop_on_silence_enable;	terminate recording on silence (record only)
unsigned char	stop_on_noise_enable;	terminate recording on noise (record only)

}VrSetPortParameters, *pVrSetPortParameters;

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This multi-purpose function is used to set the port parameter table settings for a given voice resource channel.

Returns
This function will return a XDS_SUCCESS (0) if successful or a non-zero return value indicates an error condition:

ILL_ARG	**-** the VrSetPortParameters structure passed to the library is NULL
XDS_WRONG_PARAMETER	– the enable mode for playback and/record was invalid
XDS_WRONG_BOARD	– a board other than an XDS Voice Resource board was selected
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL()	- an IOCTL was called and returned a value from the IOCTL call

Comments
This user function is to be used in conjunction with other user library functions and will only set port specific parameters on the board. When the user wishes to enable options such as the AGC (for record), they will need to set the global parameters for AGC first (using the xds_vr_set_port_parameters() XDS library function first).

The options for **tag_value** are:

SET_PLAY_GAIN	- tells the function to set the playback gain
SET_RECORD_GAIN	- tells the function to set the record gain
SET_PLAY_DTMF_STOP_BITMMAP	- tells the function the bitmap of the DTMF digit(s) to stop playback on
SET_RECORD_DTMF_STOP_BITMAP	- tells the function the bitmap of the DTMF digit(s) to stop recording on
SET_RECORD_SILENCE_STOP_ENABLE	- tells the function to enable stop record on silence
SET_RECORD_NOISE_STOP_ENABLE	- tells the function to enable stop record on noise
SET_AGC_ENABLE	- tells the function to enable and set the record AGC parameters

To set multiple port parameters to be set, the **tag_value** options can be 'ored' together (ie: SET_PLAY_GAIN | SET_PLAY_DTMF_STOP_BITMMAP will set both play gain and stop playback on DTMF parameters) .

The value for **play_dtmf_terminate_bitmap** is the digit map for stop on DTMF digit for playback. Each bit represents a DTMF digit. If the bit is set, stop on DTMF is enabled for that digit. If a bit is clear, stop on DTMF is disabled for that digit. This function is only active if DTMF detection is enabled for the port. All digits are reported to the application regardless of the setting of this parameter. The bits are defined as follows: Bits 0 through 9 correspond to digits 0 through 9. Bits 10 through 13 correspond to digits A through D. Bit 14 is for '*'. Bit 15 is for "#".

The value for **record_dtmf_terminate_bitmap** is the digit map for stop on DTMF digit for record. Each bit represents a DTMF digit. If the bit is set, stop on DTMF is enabled for that digit. If a bit is clear, stop on DTMF is disabled for that digit. This function is only active if DTMF detection is enabled for the port. All digits are reported to the application regardless of the setting of this parameter. The bits are defined as follows: Bits 0 through 9 correspond to digits 0 through 9. Bits 10 through 13 correspond to digits A through D. Bit 14 is for '*'. Bit 15 is for "#".

The options for **play_gain** are:
0 – 20 (in dB steps)
Gain value for playback in dB. This is a twos complement number. The gain is always an absolute value. The maximum gain is +20 dB.

The options for **record_gain** are:
0 – 20 (in dB steps)
Gain value for recording in dB. This is a twos complement number. The gain is always an absolute value. The maximum gain is +20 dB.

The options for **agc_enable** are:

ENABLE	- this will enable AGC for recording
DISABLE	- this will disable AGC for recording

The options for **stop_on_silence_enable** are:

ENABLE	- this will enable recording termination if silence is detected on a voice resource channel for a chosen duration of time
DISABLE	- this will disable recording termination if silence is detected on a voice resource channel

The options for **stop_on_noise_enable** are:

ENABLE	- this will enable recording termination if noise is detected on a voice resource channel for a chosen duration of time
DISABLE	- this will disable recording termination if noise is detected on a voice resource channel

Examples

```
VrSetPortParameters.tag_value = SET_PLAY_GAIN;
VrSetPortParameters.play_gain = 20;

xds_vr_set_port_parameters(16, 0, &VrSetPortParameters);
```

This will enable and set the playback gain on voice resource channel 0 of board 16 to 20 dB.

```
VrSetPortParameters.tag_value = SET_RECORD_GAIN | SET_RECORD_SILENCE_STOP_ENABLE;
VrSetPortParameters.record_gain = 10;

xds_vr_set_port_parameters(16, 0, &VrSetPortParameters);
```

This will enable and set the recording gain on voice resource channel 0 of board 16 to 10 dB. It will also enable recording termination on silence (which the max silence time will need to be set using the xds_vr_set_global_parameters() XDS library function first).

xds_vr_set_global_parameters

xds_vr_set_port_parameters(board_number, &VrSetGlobalParameters);

unsigned char board_number;	a value indicating which board the command is for

typedef struct VRSETGLOBALPARAMETERS{

unsigned long	tag_value;	which parameters to be set in this function
unsigned short	silence_threshold;	silence threshold (in dBm) (record only)
unsigned short	max_silence_time;	maximum silence time before recording terminates (in seconds) (record only)
unsigned short	max_noise_time;	maximum noise time before recording terminates (in seconds) (record only)
unsigned short	agc_target_level;	target output level for AGC (in dBm) (record only)
unsigned short	agc_target_hysteresis;	target output level hysteresis for AGC (in 0.1 dBm) (record only)
unsigned short	agc_cutoff_level;	AGC cutoff level (in dBm) (record only)
unsigned short	agc_cutoff_hysteresis;	AGC cutoff level hysteresis (in 0.1 dBm) (record only)
unsigned short	agc_max_gain;	AGC maximum gain (in dB) (record only)
unsigned short	agc_gain_rate;	AGC gain increase rate (in dB/S) (record only)

}VrSetGlobalParameters, *pVrSetGlobalParameters;

Applicable boards
Infinity Series H.100 boards with a voice resource module

Purpose
This multi-purpose function is used to set global parameters.

Command Sent
0x00000009

Returns
This function will return a XDS_SUCCESS (0) if successful or a non-zero return value indicates an error condition:

ILL_ARG	**-** the VrSetGlobalParameters structure passed to the library is NULL
ILL_OPTION	– an invalid parameter option was passed to the function
XDS_WRONG_PARAMETER	– the AGC max gain and/or gain rate is out of range
XDS_WRONG_BOARD	– a board other than an XDS Voice Resource board was selected
XDS_ILL_VOICE_CHANNEL	- an invalid voice resource channel has been selected
IOCTL()	- an IOCTL was called and returned a value from the IOCTL call

Comments
This user function is to be used in conjunction with other user library functions and will only set global parameters on the board.

The options for **tag_value** are:

SET_SILENCE_THRESHOLD	- tells the function to set the silence threshold
SET_MAX_SILENCE_TIME	- tells the function to set the max silence time
SET_MAX_NOISE_TIME	- tells the function to set the max noise time
SET_AGC_TARGET_LEVEL	- tells the function to set the target output level for AGC
SET_AGC_TARGET_LEVEL_HYSTERESIS	- tells the function to set the target output level hysteresis for AGC
SET_AGC_CUTOFF_LEVEL	- tells the function to set the AGC cutoff level
SET_AGC_CUTOFF_LEVEL_HYSTERESIS	- tells the function to set the AGC cutoff level hysteresis
SET_AGC_MAX_GAIN	- tells the function to set the AGC maximum gain
SET_AGC_RATE	- tells the function to set the AGC gain increase rate

To set multiple port parameters to be set, the **tag_value** options can be 'ored' together
(ie: SET_SILENCE_THRESHOLD | SET_MAX_SILENCE_TIME will set values for both the silence threshold and max silence time) .

The options for **silence_threshold** are:
0 – 0xFFFF (in dBm)
Silence threshold in dBm. The actual threshold is –ABS(value). Thus, entering 0x0010 or 0xFFF0 will both result in a threshold of –16 dBm. The default value is –35 dBm.

The options for **max_silence_time** are:
0 - ~229 (in seconds)
Maximum silence time before stopping a record session in seconds. Default is 10 seconds. Maximum is approximately 229 seconds (values greater than this will saturate at app. 229 seconds).

The options for **max_noise_time** are:
0 - ~229 (in seconds)
Maximum noise time before stopping a record session in seconds. Default is 10 seconds. Maximum is approximately 229 seconds (values greater than this will saturate at app. 229 seconds).

The options for **agc_target_level** are:
0 – 0xFFFF (in dBm)
Target output level for AGC in dBm. The actual target is –ABS(Value). Thus, entering 0x0010 or 0xFFF0 will both result in a target of –16 dBm. The default value is –15 dBm. The AGC algorithm will decrease the signal gain whenever the output level is higher than the AGC target level.

The options for **agc_target_hysteresis** are:
0 – 0xFFFF (in 0.1 dB)
Target output level hysteresis for AGC in 0.1 dB. The actual hysteresis is –ABS(Value/10). Thus, entering 0x0010 or 0xFFF0 will both result in a target hysteresis of –1.6 dB. The default value is –0.5 dB. The AGC algorithm will increase the signal gain whenever the output level is lower than (target + hysteresis) but above the cutoff value. Thus, the default values will decrease the gain whenever the output level is below –15.5 dBm.

The options for **agc_cutoff_level** are:
0 – 0xFFFF (in dBm)
AGC Cutoff level in dBm. The actual level is –ABS(Value). Thus, entering 0x0020 or 0xFFE0 will both result in a cutoff of –32 dBm. The default value is –40 dBm. The AGC algorithm will increase the signal gain whenever the input level is higher than the cutoff level and the output level is lower than the (target + hysteresis) level.

The options for **agc_cutoff_hysteresis** are:
0 – 0xFFFF (in 0.1 dB)
AGC Cutoff level hysteresis for AGC in 0.1 dB. The actual hysteresis is –ABS(value/10). Thus, entering 0x0010 or 0xFFF0 will both result in a cutoff hysteresis of –1.6 dB. The default value is –1.0 dB. The AGC algorithm will not compute new gain levels and will insert unity gain whenever the input level is lower than (cutoff level + cutoff hysteresis). Thus, the default values will disable the AGC whenever the input level is lower than –41 dBm.

The options for **agc_max_gain** are:
0 – 20 (in dB)
AGC maximum gain in dB. The maximum gain that the AGC will insert is ABS(value). The default is 20 dB and the maximum value for this field is 20 dB.

The options for **agc_gain_rate** are:
0 – 20 (in dB / S)
AGC gain increase rate in dB/S. Sets the maximum rate of increase for gain for AGC. The gain rate will be ABS(value) dB/S. The default is 5 dB/S. The maximum is 20 dB/S.

Examples
```
VrSetGlobalParameters.tag_value = SET_MAX_SILENCE_TIME;
VrSetGlobalParameters.max_silence_time = 5;

xds_vr_set_global_parameters(16, &VrSetGlobalParameters);
```

This will set the maximum silence time for record for all voice channels on board 16 to 5 seconds. The user will need to tell the port to enable stop on silence for the voice resource channel by calling the xds_vr_set_port_parameters() XDS user library function.

```
VrSetGlobalParameters.tag_value = SET_SILENCE_THRESHOLD | SET_MAX_SILENCE_TIME;
VrSetGlobalParameters.max_silence_time = 5;
VrSetGlobalParameters.silence_threshold = 0x0010;

xds_vr_set_global_parameters(17, &VrSetGlobalParameters);
```

This will set the maximum silence time for record for all voice channels on board 17 to 5 seconds with a silence threshold of –16 dBm. The user will need to tell the voice channel to enable stop on silence for the voice resource channel by calling the xds_vr_set_port_parameters() XDS user library function.

H.100 / H.110 MVIP-compatibility Commands

This page was intentionally left blank.

xds_ct_mvip_conference_det_dtmf

xds_ct_mvip_conference_det_dtmf(board_number, cca, mode, time);

unsigned char board_number;	a value indicating which board the command is for
int cca;	a value indicating which conference to enable or disable detection on
char mode;	a value of either 'D' or 'E', indicating which mode to use for detection
int time;	a value between 0 and 0xCF determining the duration of tone clamping to be used

Applicable boards
H.100/H.110 boards with conferencing resources and the MVIP MC3 / Enhanced Conference board

Purpose
This function is used to enable or disable DTMF detection on a "CCA".

Message Sent
"MDcccmdd" where **ccc** is the CCA of the conference, **m** is the detection mode, and the **dd** is the duration of detection.
or
"MDccmdd" where **cc** is the CCA of the conference, **m** is the detection mode, and the **dd** is the duration of detection.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid detection mode was chosen
ILL_ARG	an illegal clamping time value was selected
ILL_CCA	an illegal CCA value was used for the board
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This is the "MVIP" style command for enabling DTMF detection on a specified "CCA".
xds_ct_conference_dtmf is also available for use.

Example

xds_ct_mvip_conference_det_dtmf(16, 14, 'E', 0);	this will disable DTMF detection on timeslot 14 from the mix using handle 4 on board 16.

xds_ct_mvip_conference

int xds_ct_mvip_conference(board_number, handle, cca, mode, atten, threshold);

unsigned char board_number;	a value indicating which board the command is for
int handle;	a value indicating the conference number
int cca;	the selected conference control address to be used
char mode;	the conference mode to be used
char atten;	a value between 0-7 that controls the input and output attenuation of the conference on the specified timeslots
char threshold;	a value between 0-3 specifying a noise threshold below which the input will not be added to a conference

Applicable Boards

H.100/H.110 boards with conferencing resources

Purpose

This will create an MVIP compliant conference command and send it to the board.

Returns

This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ATTEN	invalid attenuation value was chosen
ILL_THRESHOLD	invalid threshold value was chosen
ILL_HANDLE	conference handle out of range was chosen
ILL_ARG	invalid conference mode or combination of CCA and handle is illegal
ILL_MODE	invalid conference mode
ILL_CCA	conference control address out of range was chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Message Sent

MKhhccmat – where the **hh** is the conference handle to be used, **cc** is the CCA, **m** is the mode, **a** indicates the attenuation value, and the **t** is the threshold value.

Comments
For a more detailed explanation of any of the conferencing limitations of your board, please consult the board's reference manual. A chart of available combinations of CCAs and handles is available on page 7-3 of the "*Infinity Series H.110 512/256 Port Conference Board*" manual (258M006). This will give the user an idea of how the conference blocks are set up. The following is a brief list of valid parameter ranges:

Example

xds_ct_mvip_conference(16, 127, 384, 'E', 6, 2);	this will enable conferencing on board 16 with a handle of 127 and CCA of 384, with attenuation will be set to a value of 6 and the threshold set to 2

xds_ct_mvip_conference_output_disable

int xds_ct_mvip_conference_output_disable(board_number, output_bus,
output_stream, output_timeslot);

unsigned char board_number;	a value indicating which board the command is for
char output_bus;	a value indicating which bus to control
int output_stream;	the selected output stream
int output_timeslot;	the selected output timeslot

Applicable Boards
H.100/H.110 boards with conferencing resources

Purpose
This will disable output in a conference on the selected MVIP-95 terminus.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	illegal output bus chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Message Sent
MObssttD **–** where the **b** is the terminus bus, **ss** is the stream, and **tt** is the timeslot.

Comments
This function can be used to send an MVIP-95 compliant conference disable message to an XDS Infinity Series conference board. For information on the MVIP-95 terminus and valid values, please refer to the MVIP-95 specification.

Example

xds_ct_mvip_conference(16, 'L', 0, 3);	this will disable the conference output on stream 0, timeslot 3 of the local bus on board 16

xds_ct_mvip_conference_output_enable

int xds_ct_mvip_conference_output_enable(board_number, output_bus,
output_stream, output_timeslot, input_bus, input_stream, input_timeslot);

unsigned char board_number;	a value indicating which board the command is for
char output_bus;	the output bus
int output_stream;	the selected output stream
int output_timeslot;	the selected output timeslot
char input_bus;	the input bus
int input_stream;	the selected input stream
int input_timeslot;	the selected input timeslot

Applicable Boards
H.100/H.110 boards with conferencing resources

Purpose
This will enable a conference on the selected CT bus, stream, and timeslot

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	illegal input and/or output bus chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Message Sent
MObssttEbsstt – where the first **b** is the output terminus bus, first **ss** is the output stream, and first **tt** is the output timeslot, and the second **b** is the input terminus bus, second **ss** is the input stream, and second **tt** is the input timeslot.

Comments
This function can be used to send an MVIP-95 compliant conference enable message to an XDS Infinity Series conference board. For information on the MVIP-95 terminus and valid values, please refer to the MVIP-95 specification.

Example
xds_ct_mvip_conference_output_enable(16, ‘L’, 0, 3, ‘H’, 0, 4);

This will conference input stream 0, timeslot 4 on the H.110 bus with the output stream 0, timeslot 3 of the local bus on board 16.

xds_ct_mvip_conference_output_pattern

**int xds_ct_mvip_conference_output_pattern(board_number, output_bus,
output_stream, output_timeslot, pattern);**

unsigned char board_number;	a value indicating which board the command is for
char output_bus;	the output bus
int output_stream;	the selected output stream
int output_timeslot;	the selected output timeslot
int pattern	a pattern value

Applicable Boards
H.100/H.110 boards with conferencing resources

Purpose
This will output a pattern value on a selected MVIP-95 terminus.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	illegal input and/or output bus chosen
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Message Sent
MObssttPpp – where the **b** is the output terminus bus, **ss** is the output stream, and **tt** is the output timeslot, and **pp** is the pattern value to output

Comments
This function can be used to send an MVIP-95 compliant output pattern message to an XDS Infinity Series conference board. For information on the MVIP-95 terminus and valid values, please refer to the MVIP-95 specification.

Example
xds_ct_mvip_conference_output_enable(16, ‘L’, 0, 3, 55);

This will output the pattern “55” on output stream 0, timeslot 3 on the local bus of board 16.

xds_ct_mvip_line_det_dtmf

xds_ct_mvip_line_det_dtmf(board_number, detector, mode);

unsigned char board_number;	a value indicating which board the command is for
int detector;	the detector number to control
char mode;	a value of either 'D' or 'E', indicating which mode to use for detection

Applicable boards
H.100/H.110 line interface boards

Purpose
This function is used to enable or disable DTMF detection on a line board.

Message Sent
"MDhhd" where **hh** is the detector and d is the mode of detection.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid detection mode was chosen
ILL_ARG	an illegal detector was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This is the "MVIP" style command for enabling DTMF detection on a detector.

Example

xds_ct_mvip_line_det_dtmf(1, 2, 'E');	this will enable DTMF detection on detector 2 on board 1.
xds_ct_mvip_line_det_dtmf(1, 2, 'D');	this will disable DTMF detection on detector 2 on board 1.

xds_ct_mvip_detect_energy

xds_ct_mvip_detect_energy(board_number, detector, duration);

unsigned char board_number;	a value indicating which board the command is for
int detector;	the detector number to control
int duration;	a value from 0 to 223

Applicable boards
H.100/H.110 line interface boards

Purpose
This function is used to enable or disable energy detection on a line board.

Message Sent
"MEhhdd" where **hh** is the detector and dd is the duration of detection.
or
"MEhhF" where **hh** is the detector (disable mode).

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid detection duration was selected
ILL_ARG	an illegal detector was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This is the "MVIP" style command for enabling energy detection on a detector.

A value of 0 for the duration will disable the detection and values from 1 to 223 (in .1 second increments) will enable it.

Example

xds_ct_mvip_detect_energy(1, 2, 220);	this will enable energy detection on detector 2 on board 1 for 220 milliseconds.
xds_ct_mvip_detect_energy(1, 2, 0);	this will disable energy detection on detector 2 on board 1.

xds_ct_mvip_send_dial_string

xds_ct_mvip_detect_energy(board_number, generator, dial_string);

unsigned char board_number;	a value indicating which board the command is for
int generator;	the generator number to control
char *dial_string;	the dial string of digits to be sent

Applicable boards
H.100/H.110 line interface boards

Purpose
This function is used to enable or disable DTMF detection on a line board.

Message Sent
"MGhh(digits)" where **hh** is the generator and (**digits**) is the string of digits to be sent.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_ARG	an illegal generator was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This is the "MVIP" style command for enabling energy detection on a detector.

Example
xds_ct_mvip_detect_energy(1, 2, "0123");

this will send the digits "0123" out from generator 2 on board 1 for 220 milliseconds.

xds_ct_mvip_conference_set_gain

xds_ct_mvip_conference_set_gain(board_number, cca, polarity, gain);

unsigned char board_number;	a value indicating which board the command is for
int cca;	a value indicating which conference to enable or disable detection on
char polarity;	a value of either '-' or '+', indicating the sign of the polarity for the gain
char gain;	a value between 0 and 0xF

Applicable boards
H.110 512 port conference board

Purpose
This function is used to set the output gain and polarity of a conference.

Message Sent
"MGcccab" where **ccc** is the CCA of the conference, **a** is the sign of the polarity, and the **b** is the value of the gain.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid polarity sign was chosen
ILL_ARG	an illegal gain value was selected
ILL_CCA	an illegal CCA value was used for the board
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This is the “MVIP” style command for setting the gain parameters on a conference. This will be used to set the output gain of conferences created using the “MK” MVIP-compatible command. Note that in the ccc argument, the last two digits are the same as used in the MK command and the first digit is 0 if handles 01-54 were used and 1 if handles 55-A8 were used.

Legal values for the polarity and gain are:

+0x1 to +0xF (+3 dB to +45 dB)
±0x0 (0 dB)
-0x1 to -0xE (-3 dB to -42 dB)
-0xF (-45 dB (silence))

Example

xds_ct_mvip_conference_set_gain(2, 1, ‘+’, 3);	this will set the output gain on CCA 1 to +9 dB on board 2.
xds_ct_mvip_conference_set_gain(2, 1, ‘-’, 1);	this will set the output gain on CCA 1 to -3 dB on board 2.

xds_ct_mvip_conference_control_gain

xds_ct_mvip_conference_control_gain(board_number, cca, polarity, gain);

unsigned char board_number;	a value indicating which board the command is for
int cca;	a value indicating which conference to enable or disable detection on
char polarity;	a value of either '-' or '+', indicating the sign of the polarity for the gain
char gain;	a value between 0 and 0xF

Applicable boards
H.110 512 port conference board

Purpose
This function is used to control the input gain and polarity of a conference.

Message Sent
"MIcccab" where **ccc** is the CCA of the conference, **a** is the sign of the polarity, and the **b** is the value of the gain.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an invalid polarity sign was chosen
ILL_ARG	an illegal gain value was selected
ILL_CCA	an illegal CCA value was used for the board
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This is the “MVIP” style command for setting the gain parameters on a conference. This will be used to control the gain of conferences created using the “MK” MVIP-compatible command. Note that in the ccc argument, the last two digits are the same as used in the MK command and the first digit is 0 if handles 01-54 were used and 1 if handles 55-A8 were used.

Legal values for the polarity and gain are:

+0x1 to +0xF (+3 dB to +45 dB)
±0x0 (0 dB)
-0x1 to -0xE (-3 dB to -42 dB)
-0xF (-45 dB (silence))

Example

xds_ct_mvip_conference_control_gain(2, 1, ‘+’, 3);	this will set the gain on CCA 1 to +9 dB on board 2.
xds_ct_mvip_conference_control_gain(2, 1, ‘-’, 1);	this will set the gain on CCA 1 to -3 dB on board 2.

xds_ct_mvip_dial_pulse_detect

xds_ct_mvip_dial_pulse_detect(board_number, port, mode);

unsigned char board_number;	a value indicating which board the command is for
int port;	the port to detect pulse on
char mode;	the detection mode to be used, ‘E’ for enable, ‘D’ for disable

Applicable boards
H.100/H.110 line interface boards

Purpose
This function is used to enable or disable dial pulse detection on a line board.

Message Sent
“MPxxm” where **xx** is the port and **m** is the detection mode.

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an illegal detection mode was selected
ILL_PORT	an illegal port was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This is the “MVIP” style command for enabling pulse detection on a port.

Example
xds_ct_mvip_dial_pulse_detect(1, 2, ‘E’);

this will enable dial pulse detection on port 2 of board 1.

xds_ct_mvip_dial_pulse_detect(1, 2, ‘D’);

this will disable dial pulse detection on port 2 of board 1.

xds_ct_mvip_tristate

xds_ct_mvip_tristate(board_number, mode);

unsigned char board_number;	a value indicating which board the command is for
char mode;	the tristate mode to be used, 1 for enable, 0 for disable

Applicable boards
H.100/H.110 line interface boards

Purpose
This function is used to enable or disable dial pulse detection on a line board.

Message Sent
“MTD”
or
“MTE”

Returns
This function will return a 0 if successful. A non-zero return value indicates an error condition:

ILL_MODE	an illegal tristate mode was selected
IOCTL	an IOCTL was called and returns a return value from the IOCTL call

Comments
This is the “MVIP” style command for tristating the CT bus.

Example
xds_ct_mvip_tristate(1, 0);

this will disable CT bus output on board 1.

xds_ct_mvip_tristate(1, 1);

this will enable CT bus output on board 1.

This page was intentionally left blank.

Receive Message Types
(SCSA Library Version)

This page was intentionally left blank.

Received Message Types

Unknown Message
Unknown Message - an unknown message type was received from a board
msg_type = 0, msg_sub = 0, port_number = -1, timeslot = -1, argument = 0, msg_str = "U(message)"

State Change Messages (Type 1 - Line Boards Only)

Busy Port - Port found busy in response to a **xds_seize**
msg_type = 1, msg_sub = 1, port_number = port, timeslot = -1, argument = 0, msg_str = message

Connect Acknowledge - Port placed in the connect state by **xds_connect**
msg_type = 1, msg_sub = 2, port_number = port, timeslot = -1, argument = 0, msg_str = message

DID - DID address digits detected on a port
msg_type = 1, msg_sub = 3, port_number = port, timeslot = -1, argument = 0,
msg_str = address digit string

Tone String Ended - A DTMF string sent by **xds_send_dtmf** has completed
msg_type = 1, msg_sub = 4, port_number = port, timeslot = -1, argument = 0, msg_str = message

Off-Hook - a port has been detected going off hook
msg_type = 1, msg_sub = 5, port_number = port, timeslot = -1, argument = 0, msg_str = message

Hold - a port has been put in the hold state by **xds_hold**
msg_type = 1, msg_sub = 6, port_number = port, timeslot = -1, argument = 0, msg_str = message

Idle - a port has been put in the idle state by **xds_disconnect**
msg_type = 1, msg_sub = 7, port_number = port, timeslot = -1, argument = 0, msg_str = message

Conference - a port has been conferenced using the local facilities
msg_type = 1, msg_sub = 8, port_number = port, timeslot = -1, argument = 0, msg_str = message

Listen Acknowledge - a port has been placed in the listen state by **xds_audit** or **xds_listen_dtmf**
msg_type = 1, msg_sub = 9, port_number = port, timeslot = -1, argument = 0, msg_str = message

On-Hook - a port has been detected going on-hook while seized or connected
msg_type = 1, msg_sub = 10, port _number = port, timeslot = -1, argument = 0,
msg_str = message

Hook-Flash- a hook flash has been detected on a station or E&M port
msg_type = 1, msg_sub = 11, port_number = port, timeslot = -1, argument = 0, msg_str = message

Ring Detect - ringing has been detected on a loop or ground start port
msg_type = 1, msg_sub = 12, port_number = port, timeslot = -1, argument = cadence (in ASCII), msg_str = message

DTMF tone - a DTMF tone has been detected on a port set to listen by **xds_listen_dtmf**
msg_type = 1, msg_sub = 13, port_number = port, timeslot = -1, argument = tone (in ASCII), msg_str = message

DTMF tone - a DTMF tone has been detected on a port set to listen by **xds_conference_dtmf**
msg_type = 1, msg_sub = 13, port_number = -1, timeslot = SCbus timeslot,
argument = tone (in ASCII), msg_str = message

Transmit Acknowledge - a port has been placed in the transmit state by **xds_transmit**
msg_type = 1, msg_sub = 14, port_number = port, timeslot = -1, argument = 0,
msg_str = message

Energy Detect - energy has been detected on a port set to detect by **xds_detect_energy**
msg_type - 1, msg_sub = 15, port_number = port, timeslot = -1,
argument = 1 if energy detected, 0 if energy stops, msg_str = message

Unknown - unknown state change message type
msg_type = 1, msg_sub = 0, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Acknowledgements

Reset All - acknowledges a **xds_reset_all**
msg_type = 2, msg_sub = 1, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Interrupt On - ackowledges a **xds_msg_on**
msg_type = 2, msg_sub = 2, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Clock Mode Set - the clock mode on the SCbus has been set
msg_type = 2, msg_sub = 3, port_number = -1, timeslot = -1, argument = clock mode (in ASCII), msg_str = message

Next Clock Set - the board has been set to supply Next Clock on a clock failure
msg_type = 2, msg_sub = 4, port_number = -1, timeslot = -1,
argument = next clock mode (in ASCII), msg_str = message

Clock Mode for Secondary Bus Set - the clock mode of the secondary bus of conference board has been set
msg_type = 2, msg_sub = 5, port_number = -1, timeslot = -1, argument = clock mode (in ASCII), msg_str = message

Next Clock for Secondary Bus Set - the board has been set to supply Next Clock to the secondary bus on a clock failure
msg_type = 2, msg_sub = 6, port_number = -1, timeslot = -1,
argument = next clock mode (in ASCII), msg_str = message

DSP Reset - acknowledges an **xds_reset_dsp**
msg_type = 2, msg_sub = 7, port_number = -1, timeslot = -1, argument = 0,
msg_str = text of message

Clock Errors

Clock Failure - a clock failure has been detected
msg_type = 3, msg_sub = 1, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Clock Restored - a clock restoration has been detected
msg_type = 3, msg_sub = 2, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Next Clock Assumed - the board has become the clock master after detecting a clock failure
msg_type = 3, msg_sub = 3, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Write Error Due to Lack of Clock - a switching error due to a lack of clock was detected
msg_type = 3, msg_sub = 4, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Clock Failure on Secondary Bus - a clock error on the secondary bus of a conference board
msg_type = 3, msg_sub = 5, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Clock Restored on Secondary Bus - a clock restoration on the secondary bus of a conference board
msg_type = 3, msg_sub = 6, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Next Clock Assumed on Secondary Bus - the board has become clock master on the secondary bus after detecting a clock failure
msg_type = 3, msg_sub = 7, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Write Error Due to Lack of Clock on Secondary Bus - a switching error due to lack of a clock on the secondary bus of a conference board
msg_type = 3, msg_sub = 8, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Memory Error Detected Due to Clock - a switching memory error due to a clock error has been detected on a conference board
msg_type = 3, msg_sub = 9, port_number = -1, timeslot = -1, argument = 0, msg_str = message

SCxbus Clock Failure - a clock failure on the SCx bus has been detected
msg_type = 3, msg_sub = 10, port_number = -1, timeslot = -1, argument = 0, msg_str = message

SCxbus Clock Restored - a clock restoration on the SCx bus has been detected
msg_type = 3, msg_sub = 11, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Master Clock Attempt Failure - an attempt to become the Master Clock has failed because the CLKFAIL line was low, (an EC8 or EC08) message
msg_typ = 3, msg_sub = 12, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Master Clock Attempt Failure on secondary drive - an attempt to become the Master Clock has failed because the CLKFAIL line was low, (an EC18) message
msg_typ = 3, msg_sub = 13, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Argument Errors

Timeslot Error - an illegal timeslot number was used in a message
msg_type = 4, msg_sub = 1, port_number = -1, timeslot = illegal timeslot, argument = 0,
msg_str = timeslot string in message

Pattern Error - an illegal pattern value was used in a message
msg_type = 4, msg_sub = 2, port_number = -1, timeslot = -1, argument = 0,
msg_str = illegal pattern string

Illegal Clock Mode - an illegal clock mode value was used in a message
msg_type = 4, msg_sub = 3, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Illegal Next Clock Mode - an illegal next clock mode value was used in a message
msg_type = 4, msg_sub = 4, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Path Conflict Error - an MVIP connection could not be made on an SCSA/MVIP Bridge Board due to a switching conflict
msg_type = 4, msg_sub = 5, port_number = -1, timeslot = -1, argument = 0,
msg_str = MVIP timeslot string in message

Line Board Resource Error

No DTMF Detector - All DTMF detectors were in use when one was requested
msg_type = 5, msg_sub = 1, port_number = port, timeslot = -1, argument = 0, msg_str = message

No DTMF Generator - all DTMF generators were in use when one was requested
msg_type = 5, msg_sub = 2, port_number = port, timeslot = -1, argument = 0, msg_str = message

DTMF DID Timeout - a timeout occured while expecting DTMF DID address digits
msg_type = 5, msg_sub = 3, port_number = port, timeslot = -1, argument = 0, msg_str = message

Wink Detect Timeout - a timeout occured while expecting a wink on an outgoing call
msg_type = 5, msg_sub = 4, port_number = port, timeslot = -1, argument = 0, msg_str = message

Transmit Backplane Level - the transmit backplane level exceeds the capabilities of the board
msg_type = 5, msg_sub = 6, port_number = port, timeslot = -1, argument = 0, msg_str = message

Receive Backplane Level - the receive backplane level exceeds the capabilities of the board
msg_type = 5, msg_sub = 7, port_number = port, timeslot = -1, argument = 0, msg_str = message

Transmit Level Error - the transmit level exceeds the capabilities of the board
msg_type = 5, msg_sub = 8, port_number = port, timeslot = -1, argument = 0, msg_str = message

Receive Level Error - the receive level exceeds the capabilities of the board
msg_type = 5, msg_sub = 9, port_number = port, timeslot = -1, argument = 0, msg_str = message

Conference Board Errors

Handle Error - an illegal handle number was used
msg_type = 6, msg_sub = 1, port_number = -1, timeslot = -1, argument = handle,
msg_str = message

No Conference Control Address - no conference control address was available for the conference
msg_type = 6, msg_sub = 2, port_number = -1, timeslot = -1, argument = handle,
msg_str = message

No Internal Path for Destination - no internal switch path was available for a destinatiion
msg_type = 6, msg_sub = 3, port_number = -1, timeslot = destination timeslot, argument = 0,
msg_str = message

No Internal Path for Source - no internal switch path was available for a source
msg_type = 6, msg_sub = 4, port_number = -1, timeslot = source timeslot, argument = 0,
msg_str = message

Diagnostic Responses

DSP Message - a message from a line board DSP
msg_type = 7, msg_sub = 1, port_number = -1, timeslot = -1, argument = 0,
msg_str = DSP message

Unparseable Messages

Unparseable Messages - an unparseable message was received by the board
msg_type = 8, msg_sub = 1, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Basic Rate ISDN Board Messages

Layer 3 Message Received - an LC or LR message received with a layer 3 portion
msg_type = 9, msg_sub = 1, port_number = port, timeslot = SAPI, argument = TEI, msg_str = message in main mailbox, augRxLen = length of Layer 3 message, augRxMesg = body of Layer 3 message

'D' Message Received - A 'D' format layer 3 call control message
msg_type = 9, msg_sub = 2, port_number = B_channel, timeslot = -1, argument = Layer 3 message type, msg_str = message

Layer 3 'D' Message Types

argument	**message**	**Q.931 message type**	**additional information**
1	DAxxps	ALERTing	progress indicator, signal
2	DCxxps	CONNect	connect b-channel
3	DCxxA	CONNect ACK	connect acknowledge on b-channel
4	DDxxccsrr	DISConnect	cause, signal, call reference
5	DPxxps	CALL PROCeeding	progress indicator, signal
6	DPxxPccps	PROGress	cause, progress indicator, signal
7	DRxx	RELease	release b-channel
8	DRxxccrr	RELease COMplete	cause, call reference
9	DSxxb(K)#	SETUP (overlap)	bearer capability, called number (NT only)
10	DSxxbC#	SETUP (enbloc)	bearer capability, called number (NT only)
11	DSxxbps#/#	SETUP	bearer capability, progress indicator, signal, calling #, called # (TE only)
12	DHxxrr	HOLD	call reference
13	DHxxArr	HOLD ACK	call reference
14	DHxxRccrr	HOLD REJect cause,	call reference
15	DGxxrr	RETrieve	call reference
16	DGxxArr	RETrieve ACK	call reference
17	DGxxRccrr	RETrieve REJect	cause, call reference
18	DKxxkINFO, keypad		keypad element

19	DFxxffINFO, feature key		feature key
20	DFxxffa	INFO, feature ind.	feature indicator, status
21	DIxxCxx	INFO, cause	
22	DIxxEuutt	INFO, endpoint	USID, TID (multiples?)
23	DIxxS#	INFO, SPID	SPID
24	DIxxTtext	INFO, text	text
25	DIxxLytext	INFO, line y text	text
26	DIxxQqq	INFO, request	requested item
27	DXxxccs	STATUS	cause, state
28	DNxxn	NOTIFY	notification indicator
29	DYxx	REGister	
30	DIxxFee	INFO, fujitsu	element
31	DIxxA(SPID)	AT&T MIM	SPID
32	DExxArr	KEY SETUP ACK	call reference
58	DKxxC	SENDing complete	
63	DTxxLI(text)	INFO textline	

<u>EURO ISDN Layer 3 ‘D’ Message Types</u>

argument	**message**	**Q.931 message type**	**additional information**
50	DGxxrr	RESume	call reference
51	DGxxArr	RESume ACK	call reference
52	DGxxRccrr	RESume REJect	cause, call reference
53	DHxxrr	HOLD	call reference
54	DHxxArr	HOLD ACK	call reference
55	DHxxRccrr	HOLD REJect cause,	call reference
56	DIxxKkk	INFO KEYpad	facility digits
57	DIxxccKkk	INFO KEYpad	facility digits, with cause
59	DSxxcc	SETUP	cause, call reference
60	DSxxbps#/#	SETUP	bearer capability, progress indicator, signal, calling #, called # (TE only)
61	DSxxb(K)#	SETUP (overlap)	bearer capability, called number (NT only)
62	DSxxAp	SETUP ACK	
64	DZxxA	Restart ACK	A0 = specified B-channel, A6 = both B-channels
65	DIxxcc	INFO	cause
65	DFxxcrrid(tt)	FACILITY	cause
65	DFxxFIidopb#/#	Call Forwarding (invoke)	id value, operation, procedure, basic service mode, subcriber number, forwarding number
65	DFxxFRid	Call Forwarding (return result)	id value
65	DFxxFEidee	Call Forwarding (return error)	id value, error number

Configuration Memory Response - a confirmation to a **xds_set_config()** call
msg_type = 9, msg_sub = 3, port_number = 0, timeslot = -1, argument = 0 if success, 1 failure,
msg_str = message

TEI assigned - A Terminal Endpoint Identifier assigned to a BRI port
msg_type = 9, msg_sub = 4, port_number = port, timeslot = -1, argument = TEI,
msg_str = message

TEI removed - A Terminal Endpoint Identifier Removed from a BRI port
msg_type = 9, msg_sub = 5, port_number = port, timeslot = -1, argument = TEI,
msg_str = message

TEI Query response - a response to a TEI Query message
msg_type = 9, msg_sub = 6, port_number = port, timeslot = -1, argument = 0;
msg_str = list of TEI's

Layer 1 or 2 error - a Layer 1 or Layer 2 error has been detected
msg_type = 9, msg_sub = 7, port_number = port, timeslot = -1, argument = error,
msg_str = message

Data Link Error - a Data Link Error has been detected
msg_type = 9, msg_sub = 8, port_number = port, timeslot = -1, argument = error,
msg_str = text of message

Port Reset - acknowledges an **xds_reset_port**
msg_type = 9, msg_sub = 9, port_number = port, timeslot = -1, argument = 0,
msg_str = text of message

SPID Query response - a response to a SPID Query DQxx message
msg_type = 9, msg_sub = 10, port_number = port, timeslot = -1, argument = 0,
msg_str = SPID and DN

"ETxxm"
U-Interface maintenance mode message
msg_type = 9, msg_sub = 11, port_number = port, timeslot = -1, argument = mode m,
msg_str = message

"XMxxaabb"
U-Interface monitor message
msg_type = 9, msg_sub = 12, port_number = port, timeslot = -1, argument = aa,
msg_str = message

Diagnostic - an unexpected event has generated a diagnostic message
msg_type = 9, msg_sub = 13, port_number = port, timeslot = -1, argument = error,
msg_str = text of message

<u>H.100 MC3/Conferencing Response Messages</u>

H.100 Clock Error Message - one of the clock error bits has changed
msg_type = 10, msg_sub = 1, port_number = -1, timeslot = -1, argument = error bits,
msg_str = message

MC3 Ring Failure - a failure of an MC3 bus ring has been detected
msg_type = 10, msg_sub = 2, port_number = -1, timeslot = -1, argument = ring number,
msg_str = message

MC3 Ring Recovery - the recovery of an MC3 bus ring has been detected
msg_type = 10, msg_sub = 3, port_number = -1, timeslot = -1, argument = ring number,
msg_str = message

MC3 Ring Status - one of the MC3 rings status bits has changed
msg_type = 10, msg_sub = 4, port_number = ring, timeslot = -1, argument = status bits,
msg_str = message

MC3/Conference Board Conference Error - a conference error has been detected
msg_type = 10, msg_sub = 5, port_number = handle, timeslot = -1, argument = error type,
msg_str = message

H.100 Path error - a command has failed because of the unavailability of a switching path
msg_type = 10, msg_sub = 6, port_number = -1, timeslot = destination timeslot, argument = -1,
msg_str = message

<u>H.100 E1 Response Messages</u>

Alarm Events

“ABdd”
Blue alarm (AIS) event
msg_type = 11, msg_sub = 43, port_number = span used, msg_str = message

“AGdd”
Green alarm event (no alarms active)
msg_type = 11, msg_sub = 44, port_number = span used, msg_str = message

"ARdd"
Red alarm (RCL) event
msg_type = 11, msg_sub = 45, port_number = span used, msg_str = message

"AYdd"
Yellow alarm (RAI)
msg_type = 11, msg_sub = 46, port_number = span used, msg_str = message

"AVddx"
'V' alarm
msg_type = 11, msg_sub = 47, port_number = span used, msg_str = message,
argument = 'C' or 'S'(x)

Error Messages

"ECxx"
Clock error
msg_type = 11, msg_sub = 4, msg_str = message, argument = error bit (xx)

"EDddy"
Diagnositc error
msg_type = 11, msg_sub = 5, msg_str = message, port_number = span,
argument = diagnostic error (y)

"ELddy"
Layer 1 or 2 error
msg_type = 11, msg_sub = 6, msg_str = message, port_number = span,
argument = Layer 1 / 2 error (y)

"EMddy"
MDL error (Layer 2 protocol errors)
msg_type = 11, msg_sub = 7, msg_str = message, port_number = span,
argument = MDL error (y)

"ESddss"
Line interface Unit event
msg_type = 11, msg_sub = 8, msg_str = message, port_number = span,
argument = event bits (ss)

"ETxx"
Address digit timeout
msg_type = 11, msg_sub = 9, msg_str = message, port_number = B-channel

"ETxxe"

Layer 3 timer timeout event
msg_type = 11, msg_sub = 10, msg_str = message, port_number = B-channel, argument = timeout event (e)

“EWxx”
Wink timeout
msg_type = 11, msg_sub = 11, msg_str = message, port_number = B-channel

“EXddss”
Elastic store event
msg_type = 11, msg_sub = 12, msg_str = message, port_number = span, argument = store event (ss)

Response Messages

“FBxxt”
MF-R2 backward tone detected
msg_type = 11, msg_sub = 50, msg_str = message, argument = tone detected

“FFxxt”
MF-R2 forward tone detected
msg_type = 11, msg_sub = 51, msg_str = message, argument = tone detected

Layer 3 ‘D’ Responses

“DAxxprrrr” (NT) / “DAxxprrrr(pc) (TE)”
ALERTing message
msg_type = 11, msg_sub = 13, msg_str = message, port_number = B-channel, argument = 0x01 (Q.931 message type value)

“DCxxArrrr” (NT/TE)
CONNect ACKnowledge message
msg_type = 11, msg_sub = 14, msg_str = message, port_number = B-channel, argument = 0x0F (Q.931 message type value)

“DCxxprrrr(#cnct)” (NT) / “DCxxprrrr(#cnct)(pc)” (TE)
CONNect message
msg_type = 11, msg_sub = 15, msg_str = message, port_number = B-channel, argument = 0x07 (Q.931 message type value)

“DDxxcc(p)(rrrr)” (NT) / “DDxxccp(rrrr)” (TE)
DISConnect message

msg_type = 11, msg_sub = 16, msg_str = message, port_number = B-channel, argument = 0x45 (Q.931 message type value)

“DFxxNo<name>” (NT/TE)
Facility name message
msg_type = 11, msg_sub = 17, msg_str = message, port_number = B-channel, argument = 0x62 (Q.931 message type value)

“DGxxArrrr” (NT/TE)
RETrieve ACKnowledge message
msg_type = 11, msg_sub = 18, msg_str = message, port_number = B-channel, argument = 0x33 (Q.931 message type value)

“DGxxHrrrr” (NT/TE)
RETrieve message
msg_type = 11, msg_sub = 19, msg_str = message, port_number = B-channel, argument = 0x31 (Q.931 message type value)

“DGxxRccrrrr” (NT/TE)
RETrieve REJect message
msg_type = 11, msg_sub = 20, msg_str = message, port_number = B-channel, argument = 0x37 (Q.931 message type value)

“DHxxArrrr” (NT/TE)
HOLD ACKnowledge message
msg_type = 11, msg_sub = 21, msg_str = message, port_number = B-channel, argument = 0x28 (Q.931 message type value)

“DHxxRccrrrr” (NT/TE)
HOLD REJect message
msg_type = 11, msg_sub = 22, msg_str = message, port_number = B-channel, argument = 0x30 (Q.931 message type value)

“DHxxrrrr” (NT/TE)
HOLD message
msg_type = 11, msg_sub = 23, msg_str = message, port_number = B-channel, argument = 0x24 (Q.931 message type value)

“DIxxKk(k)” (NT)
INFOrmation message - Keypad Facility
msg_type = 11, msg_sub = 24, msg_str = message, port_number = B-channel,
argument = 0x7B (Q.931 message type value)

“DIxxcc” (NT/TE)
INFOrmation message
msg_type = 11, msg_sub = 25, msg_str = message, port_number = B-channel,
argument = 0x7B (Q.931 message type value)

“DKxxk(k)” (NT)
INFOrmation message - sending complete
msg_type = 11, msg_sub = 26, msg_str = message, port_number = B-channel,
argument = 0x7B (Q.931 message type value)

“DKxxC(k)” (NT)
INFOrmation message
msg_type = 11, msg_sub = 27, msg_str = message, port_number = B-channel,
argument = 0x7B (Q.931 message type value)

“DNxxR” (TE)
NOTIFY message - call resumed
msg_type = 11, msg_sub = 28, msg_str = message, port_number = B-channel,
argument = 0x7E (Q.931 message type value)

“DNxxS” (TE)
NOTIFY message - call resumed
msg_type = 11, msg_sub = 29, msg_str = message, port_number = B-channel,
argument = 0x7E (Q.931 message type value)

“DNxxnn” (TE)
NOTIFY message
msg_type = 11, msg_sub = 30, msg_str = message, port_number = B-channel,
argument = 0x7E (Q.931 message type value)

“DPxxprrrr” (NT) / “DPxxPccp(rrrr)”
CALL PROGress message
msg_type = 11, msg_sub = 31, msg_str = message, port_number = B-channel,
argument = 0x03 (Q.931 message type value)

“DPxxPccp(rrrr)” (NT) / “DPxxp” (TE)
CALL PROCeeding message
msg_type = 11, msg_sub = 32, msg_str = message, port_number = B-channel,
argument = 0x02 (Q.931 message type value)

“DQxx(#)” (NT)
Default subscriber number message
msg_type = 11, msg_sub = 33, msg_str = message, port_number = B-channel,
argument = -1 (no Q.931 message type value)

“DRxxRcc(rrrr)” (NT) / (TE)
RELease message
msg_type = 11, msg_sub = 34, msg_str = message, port_number = B-channel,
argument = 0x4D (Q.931 message type value)

“DRxxcc(rrrr)” (NT) / (TE)
RELease COMplete message
msg_type = 11, msg_sub = 35, msg_str = message, port_number = B-channel,
argument = 0x5A (Q.931 message type value)

“DSxxAprrrr” (TE)
SETUP ACKnowledge message
msg_type = 11, msg_sub = 36, msg_str = message, port_number = B-channel,
argument = 0x0D (Q.931 message type value)

“DSxxbp(#)/(#)” (NT) / “DSxxbp#(/#)” (TE)
SETUP message
msg_type = 11, msg_sub = 37, msg_str = message, port_number = B-channel,
argument = 0x05 (Q.931 message type value)

“DSxxbp(#)C(#)” (NT) / “DSxxbp#C#” (TE)
SETUP message - sending complete
msg_type = 11, msg_sub = 38, msg_str = message, port_number = B-channel,
argument = 0x05 (Q.931 message type value)

“DSxxbp(#)” (NT)
SETUP message – overlap sending
msg_type = 11, msg_sub = 39, msg_str = message, port_number = B-channel,
argument = 0x05 (Q.931 message type value)

"DXxxccssrrrr" (NT / TE)
STATUS message
msg_type = 11, msg_sub = 40, msg_str = message, port_number = B-channel,
argument = 0x7D (Q.931 message type value)

"DZxxA(C)" (NT / TE)
RESTART ACKnowledge message
msg_type = 11, msg_sub = 41, msg_str = message, port_number = B-channel,
argument = 0x4E (Q.931 message type value)

"DZxx(C)"
RESTART message (NT)
msg_type = 11, msg_sub = 42, msg_str = message, port_number = B-channel,
argument = 0x45 (Q.931 message type value)

Voice Playback Response Messages

"PTx"
Data request response message
msg_type = 12, msg_sub = 0, port_number = DSP channel, timeslot = -1, argument = 0,
msg_str = message

"PV0dt"
DTMF digit t detected response message
msg_type = 12, msg_sub = 1, port_number = DSP channel, timeslot = -1, argument = DTMF
digit received, msg_str = message

Query Messages

Unknown Response - an unknown message type on the query queue
msg_type = 16, msg_sub = 0, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Board Version - a version request response
msg_type = 16, msg_sub = 1, port_number = -1, timeslot = -1, argument = 0, msg_str = message

DSP Version - a DSP version request response
msg_type = 16, msg_sub = 2, port_number = -1, timeslot = -1, argument = 0, msg_str = message

Transmit Timeslot - a timeslot query message for a port response
msg_type = 16, msg_sub = 3, port_number = port, timeslot = slot, argument = 0,
msg_str = message

Clock Mode - a clock mode query response
msg_type = 16, msg_sub = 4, port_number = -1, timeslot = -1, argument = mode,
msg_str = message

Clock Mode of Secondary bus - a clock mode for secondary bus response
msg_type = 16, msg_sub = 5, port_number = -1, timeslot = -1, argument = mode**,**
msg_str = message

SC2000 Data - an SC2000 query response for a line board
msg_type = 16, msg_sub = 6, port_number = -1, timeslot = -1, argument = 0, msg_str = message

FMIC Data - an FMIC query response from a Multi-Chassis board
msg_type = 16, msg_sub = 7, port_number = -1, timeslot = -1, argument = 0, msg_str = message

PEB2047 Data - a PEB 2047 query response from a Multi-Chassis board
msg_type = 16, msg_sub = 8, port_number = -1, timeslot = -1, argument = 0, msg_str = message

SCbus Signal - an SCbus signal query response from a Multi-Chassis board
msg_type = 16, msg_sub = 9, port_number = -1, timeslot = -1, argument = 0, msg_str = message

SCxbus Signal - an SCxbus signal query response from a Multi-Chassis board
msg_type 16, msg_sub = 10, port_number = -1, timeslot = -1, argument = 0, msg_str = message

SC2000 Data = an SC2000 query response from a Conference board
msg_type = 16, msg_sub = 11, port_number = -1, timeslot = -1, argument = 0,
msg_str = message

Alternate Segment Checksum - response to an alternate segment version request response
msg_type = 16, msg_sub = 12, port_number = -1, timeslot = -1, argument = 0, msg_str =
message

CT812 Register Data = a CT812 query response from an H.100 board
msg_type = 16, msg_sub = 13, port_number = CT812 number, timeslot = -1, argument = -1,
msg_str = message

MT90840 Receive Data = an MT90840 receive query response from an H.100 MC3 board
msg_type = 16, msg_sub = 14, port_number = chip number, timeslot = MT-90840 receive
timeslot, argument = -1, msg_str = message

MT90840 Transmit Data = an MT90840 transmit query response from an H.100 MC3 board
msg_type = 16, msg_sub = 15, port_number = chip number, timeslot = MC3 transmit timeslot,
argument = -1, msg_str = message

<u>Queue Errors</u>

Full Receive Queue - the Receive queue was full
msg_type = 17, msg_sub = 1, port_number = -1, timeslot = -1, argument = 0, msg_str = NULL
board_number = -1

Full Query Queue - the Query queue was full
msg_type = 17, msg_sub = 2, port_number = -1, timeslot = -1, argument = 0, msg_str = NULL
board_number = -1