April 1997

HDLC HARDWARE INTERFACE & SUPPORT FOR MS-DOS

Programming and Operations Manual

LIMITED USE LICENSE AGREEMENT

Sangoma Technologies Inc. provides the computer software program contained on the medium in this package (hereinafter called the Program) and licenses its use.

THE LICENSEE SHOULD CAREFULLY READ THE FOLLOWING TERMS AND CONDITIONS BEFORE ATTEMPTING TO USE THIS PRODUCT. INSERTION OF ANY OF THE DISKETTES IN THIS PACKAGE INTO ANY MACHINE INDICATES THE LICENSEE'S ACCEPTANCE OF THESE TERMS AND CONDITIONS. IF THE LICENSEE DOES NOT AGREE WITH THE TERMS AND CONDITIONS, THE LICENSEE SHOULD PROMPTLY RETURN THE PACKAGE WITHIN 15 DAYS UNUSED AND UNCOPIED IN ANY WAY SHAPE OR FORM, AND MONIES WILL BE REFUNDED.

LICENSE:

a. The purchaser of this license (hereinafter called the Licensee) is granted a personal, non-exclusive license to use the Program in accordance with the terms and conditions set out in this agreement.

b. The Program may be used only on a single computer per license granted.

c. The Licensee and the Licensee's agents and employees shall protect the confidentiality of the Program and shall not distribute or make available the Program or documentation to any third party.

d. The Licensee may copy the programs into machine readable or printed form for backup or modification purposes only in support of the Licensee's use on a single machine. The Licensee must reproduce and include the copyright notice on any copy, modification or portion merged into another program.

e. Any portion of the Program merged into or used in conjunction with another program will continue to be subject to the terms and conditions of this agreement.

f. The Licensee may not assign or transfer the license or the program to any third party without the express prior consent of Sangoma Technologies Inc.

g. The licensee acknowledges that this license is only a limited license to use the Program and documentation, and that Sangoma Technologies Inc. retains full title to the program and documentation.

h. The Licensee shall not use, copy, modify or transfer the Program or documentation or any copy, modification or merged portion, in whole or in part, except as expressly provided for in this license. If the Licensee transfers possession of any copy, modification or merged portion of the program to a third party, the license is automatically terminated under this agreement.

TERM:

The license is effective until terminated. The licensee may terminate the license at any time by destroying the Program together with all copies, modifications and merged portion in any form. The Licensee agrees upon such termination to destroy the Program together with all copies, modifications and merged portion in any form.

LIMITED WARRANTY:

The Program is provided "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the performance of the Program is with the Licensee. Should the Program prove defective, the Licensee (and not Sangoma Technologies Inc. or an authorized dealer) shall assume the entire cost of all necessary servicing, or correction. However, Sangoma Technologies Inc. warrants the diskettes on which the Program is furnished will be free of defects in materials or workmanship under normal use for a period of 90 days from the date of delivery to the Licensee. In no event will Sangoma Technologies Inc. be liable for any damages, including incidental or consequential damages arising out of the use or inability to use the Program, even if Sangoma Technologies Inc. or an authorized dealer have been advised of the possibility
of such damages, or for any claim by any other party.

The Licensee acknowledges that the Licensee has read this agreement, understands it, and agrees to be bound by its terms and conditions. The Licensee further agrees that it is the complete and exclusive statement of the agreement between the parties and supersedes any proposal or prior agreement, oral or written, and any other communications between the parties relating to the subject matter of this agreement.

Table of Contents

2. Introduction

The Sangoma S502 SDLA card is a general co-processor communication adapter capable of supporting any RS232 based communication protocol autonomously, and providing information transfer into PC work space. V.35 and X.21 interfaces are available.

This document describes the support of the HDLC protocol on the card, together with special enhancements implemented in the Sangoma HDLC code. It should be noted that this version of HDLC code provides the frame level support for the Sangoma X.25 code and the code referred to in this manual is actually X.25 code with the X.25 level disabled.
The hardware/software solution handles the link level autonomously, without
PC intervention. The PC accesses the system as required to send or receive Information frames and to recover status and statistical data..

The implementation of HDLC on the S502 adapter corresponds to the specifications of the ISO document ISO 7776: "Information processing systems - Data Communication - High-level data link control procedures- Description of HDLC LAPB-compatible DTE data link procedures", (first edition 1986-12-15).

Conventions used in this manual

Programming conventions used are as follows:

Variables described with an **0x** prefix are hexadecimal values. All other variables are decimal.

For bit mapping, the **least significant** (**low**) bit is denoted as bit **0**.

3. Hardware

General

Sangoma supports HDLC on both the S502 v 3.0 and the S502E short card.

S502 v 3.0

The Sangoma S502 v 3.0 SDLA adapter is compatible with an ISA or Microchannel bus. The hardware configuration is as follows:

POST setup for MCA version:

Copy the .ADF file provided with your software onto the setup disk for your MCA machine. Use the automatic setup. Use the address 360 (default) unless it clashes with one of the other cards in the machine.

Clock speed:

This is factory set by Jumper **JP3** on the ISA card and **JP1** on the MCA card. Do not change without consulting your Sangoma dealer.

I/O port address:

This is set by Jumpers **JP1** and **JP2** on the ISA card.

JP1	JP2	Selection
Jumpered	Jumpered	IO Address 250-252 (Hex)
Not Jumpered	Jumpered	IO Address 300-302 (Hex)
Jumpered	Not Jumpered	IO Address 350-352 (Hex)
Not Jumpered	Not Jumpered	IO Address 360-362 (Hex)☎

☎Factory default.

Memory sharing:

This is set by software writing to the appropriate ports.

Cable pinout

RS232

Function	Pin #	
TxD	2	
RxD	3	
GND	7	
RTS	4	
CTS	5	
DTR	20	
DSR	6	
DCD	8	
TxC	15	
RxC	17	
BxC	24	(On board clock source)
Power (+12v)	10	

PLEASE NOTE:
The S502A card provides a +12V source on Pin 10 of the DB25 plug to power interface converters. Some modems or other DCE devices may ground or put voltage on Pin 10. In this case, the PC may not boot, the card may get hot or the DCE device may get hot. The solution is to disconnect the Pin 10 cable connection.

S502E card

This is a short 4 layer card, compatible with the ISA bus. Unlike the S502 v 3.0, it supports hardware interrupts as well as operating in a passive polled mode. The S502E is available with either RS232 or V.35/X.21 interfaces.

€lock speed:

This is factory set by Jumpers **JP1** and **JP2**. Do not change without consulting your Sangoma dealer.

I/O port address:

This is set by Jumper **JP3**.

Pins 1-2	Pins 3-4	Selection
Jumpered	Jumpered	IO Address 250-252 (Hex)
Not Jumpered	Jumpered	IO Address 300-302 (Hex)
Jumpered	Not Jumpered	IO Address 350-352 (Hex)
Not Jumpered	Not Jumpered	IO Address 360-362 (Hex)☎

☎Factory default.

IRQ Selection

The optional IRQ is set using **JP4**.

Pins 1-2	Pins 3-4	Pins 5-6	Pins 7-8	Selection
IN	OUT	OUT	OUT	IRQ 2
OUT	IN	OUT	OUT	IRQ 3
OUT	OUT	IN	OUT	IRQ 5
OUT	OUT	OUT	IN	IRQ 7☎

☎Factory default.

Memory sharing:

This is set by software writing to the appropriate ports.

Cable pinouts

RS232

Function	Pin #	
TxD	2	
RxD	3	
GND	7	
RTS	4	
CTS	5	
DTR	20	
DSR	6	
DCD	8	
TxC	15	
RxC	17	
BxC	24	(On board clock source)

V.35/X.21

Pin #	Function
4	RTS
5	CTS
6	DSR
7	GND
8	DCD
10	TxA
9	TxB
12	RxA
11	RxB
19	Tx Clock A
20	DTR (V10 signal)
14	DTRA (V11 signal)
13	DTRB (V11 signal)
21	Tx Clock B
22	RI
23	Rx Clock A
25	Rx Clock B
18	Aux. Clock A (On board clock source)
16	Aux. Clock B (On board clock source)

4. Software modules

General

This manual describes two interfaces which may be used to interface with the HDLC code running on the SDLA hardware:

1. The shared memory interface where the SDLA card is operated by reading and writing structures to positions in the shared memory window. Usage of this interface is described in a later section.

2. Under PC-DOS, XIP.COM may be used to provide a simple interface to avoid the complexities of direct memory accesses. The application interfaces with the XIP by means of a Software Interrupt and the XIP, in turn, reads and writes to the shared memory areas. Usage of this interface is described later.

Software support for HDLC consists of the following modules:

XLOAD.EXE loads the HDLC microcode onto the SDLA adapter, configures it, runs through a self test and starts the program.

HDLC.502 is the run time HDLC downloadable module loaded by XLOAD.EXE.

SDLA_TST.502 is a module used for testing the SDLA hardware before loading HDLC.502.

HDLC.SDL is a file defining the HDLC configuration parameters.

XIP.COM is a PC-DOS TSR program which provides a simple method of interfacing with the adapter by means of a Software Interrupt.

X25_TEST.EXE is a high-level interface to the HDLC code which allows the user to perform individual interface

commands.

XLOAD.EXE

XLOAD.EXE is a PC-DOS program and has the following command line syntax:

XLOAD -c<CODE> -f<CONFIG>

where:

CODE	is the name of the file containing the SDLA executable code for the HDLC implementation, e.g. HDLC.502.
CONFIG	is the name of the file containing the HDLC configuration information, e.g. HDLC.SDL

Example:

XLOAD -c\HDLC\HDLC.502 -f\HDLC\HDLC.SDL

XLOAD will perform a system test, read the CODE and CONFIG files and load and configure the adapter. If XLOAD does not execute successfully, an error message will be displayed and an exit code will be returned. XLOAD may also display configuration warning messages. A description of these messages and corresponding exit codes are described in Section "Error Messages".

HDLC.502

This is the HDLC support code which is loaded onto the card and establishes the link level communications.

HDLC.502 is **NOT** a PC-DOS program and is not executable under DOS.

HDLC.SDL

This file contains the HDLC code configuration data. HDLC.SDL is an ASCII text file and may be edited using any standard editor.

The default parameters are as follows:

```
io_port                 = 0x360
mem_segment             = 0xC000
mem_window              = 8
line_speed              = 9600
DTE                     = 1
T1                      = 1
T2                      = 0
T4                      = 10
N2                      = 10
k                       = 7
auto_modem_err_fixup    = 1
auto_HDLC               = 1
HDLC_config_options     = 0x00
max_I_field_length      = 1027
```

io_port

This corresponds to the base port address as set by the jumpers on the SDLA adapter. Valid values are 0x250, 0x300, 0x350 and 0x360.

mem_segment

This is the 64K PC memory segment in which the shared memory window will reside. Valid values are 0xA000, 0xC000, 0xD000 and 0xE000.

mem_window

This defines the 8k window of the PC memory segment which the PC sees as being occupied by the SDLA card. Valid values and their meanings are as follows:

Value	PC memory window
0	0000H to 1FFFH
2	2000H to 3FFFH
4	4000H to 5FFFH
6	6000H to 7FFFH
8	8000H to 9FFFH
A	A000H to BFFFH
C	C000H to DFFFH

line_speed

The HDLC link normally takes the line clocking from the modem. However, the S502 adapter will provide a clocking signal on pin 24. This means that in a local attach or test situation, there is no need for the expense or complexity of a modem eliminator. Valid values are (7.2 MHz adapter):

Value (bps)	Actual Line Speed (bps)	Comments
0	----	No clocking signal provided
1200	1202	
2400	2405	
4800	4760	
9600	9727	
19200	18643	
38400	37286	
45000	44744	
56000	55930	
64000	55930	
74447	74573	
112000	111860	
128000	111860	

DTE

Defines the station's link level and packet level configuration and should be set to **1** for a **DTE** and **0** for a **DCE**.

T1

The period timer T1 which is used for various link level retransmission and recovery procedures.

Valid values are **1** to **30** seconds and a typical value is **3** seconds.

T2

T2 is the amount of time available at the station before an acknowledging frame must be initiated in order to ensure its receipt by the remote station, prior to timer T1 running out at that station.

Valid values are **0** to **29** seconds and is generally set to **0**. This value must always be less than T1.

T4

Once the HDLC link is in the asynchronous balanced mode (ABM) information transfer phase, there is no need for any HDLC Supervisory frame traffic to occur between the two devices on the link if no Information frames are being transmitted.

The **T4** parameter is used to allow the Sangoma HDLC code to issue a link level Supervisory frame (and to elicit a response from the remote device) periodically during this 'quiescent' ABM link phase.

If this parameter is set to **0**, no unnecessary Supervisory frames will be generated. Otherwise a Supervisory frame will be issued at a period of

T4 x T1 (seconds)

For example, if the T1 timer is 3 seconds and T4 is 10, a Supervisory frame will be sent every 30 seconds.

Valid values for this parameter are **0** to **240**.

N2

This is the maximum number of transmissions and retransmissions of an HDLC frame at T1 intervals before a state change occurs.

Valid values are **1** to **30** and a typical value is **10**.

k

This parameter defines the size of the frame level window, i.e. the maximum number of sequentially numbered Information frames that may be unacknowledged at any given time.

Valid values are **1** to **7** and a typical value is **7**.

auto_modem_err_fixup

This parameter controls the reporting of modem failures to the application. If this function is enabled, then a particular modem error will only be reported once to the application. Otherwise, this error must be manually cleared by using the **SET_GLOBAL_VARIABLES** command.

Valid entries are 0 and 1, 1 enables the automatic error correction.

Note that if **auto_HDLC** is enabled, then this parameter is also automatically enabled.

auto_HDLC

If this parameter is enabled (set to **1**), then the link-level is automatically established on loading and the application need only deal with the transfer of Information frames. The CONFIGURE_LINK and HDLC_OPEN commands are not used.

The **auto_HDLC** value should only be set to **0** when the user wishes to fully control the set-up and disconnection of the HDLC link.

In general, this parameter should be set to **1** for HDLC code usage.

HDLC_config_options

This value describes the configuration options for the Sangoma HDLC code and the bit settings are as follows:

bit 0 - used for **link setup** control.
If this bit is set to **0**, then this station will issue SABM frames when link setup occurs. If set to **1**, then this station will not issues SABMs, but will react appropriately to SABMs issued by the remote station.

bit 1 - If this bit is reset, then all Information frames aborted during transmission will be immediately re-transmitted. If this bit is set, then no automatic recovery on aborts will occur and the link level protocol will be used to trigger Information frame re-transmissions.

bits 2-7 reserved.

max_I_field_length

This parameter defines the maximum number of octets in the data field of an Information frame.

Incoming frames with a data field exceeding this maximum value will instigate a Frame Reject (FRMR) response. In addition, the XIP will prevent the application from transmitting Information frames of excessive length.

Valid values are 1 to 1027 bytes.

XIP.COM

A PC-DOS interface is provided so as to avoid the complexities of direct memory access. The application interfaces with the **XIP** by means of a Software Interrupt and the **XIP**, in turn, reads and writes to shared memory area.

XIP.COM is loaded with the following command line:

XIP -f<CONFIG> -i<CALLING INTERRUPT> -r

where:

CONFIG — is the name of the file containing configuration information for the HDLC code, e.g. HDLC.SDL. **The configuration file named here must be the same as that listed in the command line arguments for XLOAD.**

CALLING INTERRUPT — is the software interrupt used by the application program to communicate with XIP. The default interrupt is **6F hex**.

The "-r" parameter is included when removing a resident XIP from memory.

e.g.

XIP -fHDLC.SDL -i7A

Load the XIP with configuration file HDLC.SDL and use the software interrupt 7A (hex) to access the XIP.

The selected software interrupt is used to activate the XIP. When executing this call, the application must set the DX and BX registers to point to a control block.

If XIP does not execute successfully, an error message will be displayed and an exit code will be returned. A description of the error messages and corresponding exit codes are described in Section "Error Messages".

X25_TEST.EXE

X25_TEST is a high-level interface to the HDLC code which allows the user to perform individual interface commands.

If X25_TEST does not execute successfully, an error message will be displayed and an exit code returned.

5. The programmer's interface

Using the HDLC Shared Memory Interface

The SDLA card is operated by reading and writing structures to positions in the shared memory window. For details of moving the structures to/from the board, see the code example later in this document.

The application program accesses the HDLC software by completing the required parameters within the control block defined below and then setting the OPP_FLAG. The SDLA processor will carry out the defined command and then update this control block with the required return code and, if applicable, the associated data buffer, data length etc. When the command has been completed, the OPP_FLAG will once again be reset.

There are two control block areas within this shared memory window:

- the SEND control block is at offset **0x16B0** from the base address of the memory window and is used for all commands except the HDLC_READ command.
- the RECEIVE control block is at offset **0x1AD0** from the base address of the memory window and is only used for the HDLC_READ command.

Both these control blocks are of the same format.

The shared memory control block structure is as follows:

Parameter	Off-set	Lth	Remarks
OPP-FLAG	00H	1	A flag set by the application to inform the SDLA processor that a COMMAND is pending. This flag is in turn reset by the processor when the COMMAND has been completed.
COMMAND	01H	1	Command code.
BUFFER_ LENGTH	02H	2	Length of the data buffer associated with this call.
RETURN_ CODE	04H	1	Result of the previous command.
PF_BIT	05H	1	The P/F bit setting.
RESERVED	06H	10	Reserved for later use.
DATA	10H	1040	This is the transfer area for passing data to and from the application level.

Using the XIP TSR Interface

The application interfaces with the XIP by means of a Software Interrupt and the XIP, in turn, reads and writes to the shared memory areas. For details of using the XIP, see the code example later in this document.

The application program will access the XIP software by means of the following commands:

MOV BX, <data segment of control block>
MOV DX, <offset of control block>

INT <CALLING INTERRUPT>

The XIP software will transfer the control block (and associated data buffer, if applicable) from the application software and carry out the defined command. It will then update this control block at the application level with the required return code and, if applicable, the associated data buffer, data length etc.

The XIP Control Block Structure is as follows:

Parameter	Off-set	Lth	Remarks
COMMAND	00H	1	Command code.
BUFFER_ LENGTH	01H	2	Length of the data buffer associated with this call.
RETURN_ CODE	03H	1	Result of the previous command.
PF_BIT	04H	1	The P/F bit setting.
RESERVED	05H	11	Reserved for later use.
DATA	10H	1040	This is the transfer area for passing data to and from the application level.

6. COMMAND Codes

The valid commands are:

0x01	CONFIGURE_LINK
0x02	HDLC_OPEN
0x03	HDLC_CLOSE
0x04	LINK_SETUP
0x05	LINK_DISCONNECT
0x06	LINK_STATUS
0x07	READ_HDLC_STATISTICS
0x08	FLUSH_HDLC_STATISTICS
0x09	READ_COMMS_ERR_STATS
0x0A	FLUSH_COMMS_ERR_STATS
0x0B	SET_GLOBAL_VARIABLES
0x0C	READ_MODEM_STATUS
0x0D	FLUSH_HDLC_DATA_BUFFERS
0x10	SEND_UNNUMBERED_INFORMATION_ FRAME
0x11	HDLC_WRITE
0x12	READ_HDLC_CONFIGURATION
0x13	SET_HDLC_CONFIGURATION
0x14	OPERATE_DATALINE_MONITOR
0x15	READ_CODE_VERSION
0x14	OPERATE_DATALINE_MONITOR
0x16	READ_TRACE_DATA
0x17	SET_INTERRUPT_TRIGGERS
0x18	READ_INTERRUPT_TRIGGERS
0x21	HDLC_READ

CONFIGURE_LINK (0x01)

This command configures the Sangoma HDLC code to use the frame level addressing corresponding to a network DTE or DCE. The link is **CLOSED** and will not transmit frames or respond to any incoming frames until an **HDLC_OPEN** has been performed. All queued Information frames are flushed, the statistic variables for the link are reset and the link is disconnected.

If 'auto_HDLC' has been enabled (set to **1**) in the configuration file, then this command should not be executed, as the station will already have been configured as defined in this configuration file.

Control Block values to be set on entry:

BUFFER_
LENGTH: Set to **0x01**.

DATA: offset 0x00 is used to define the frame level addressing as follows:

0x01 - use **DTE** addressing.

0x02 - use **DCE** addressing

Control Block values set on return:

RETURN_
CODE: 0x00 The action has been performed successfully.

0x01 The selected operational addressing mode is invalid.

0x06, 0x07, 0x08, 0x09, 0x0A
See Section "Notes on Return Codes" for further details.

HDLC_OPEN (0x02)

This command allows the HDLC code to transmit frames and to respond to any incoming frames.

DTR will be automatically raised once the HDLC_OPEN has been issued.

If 'auto_HDLC' has been enabled in the configuration file, then this command is not neccessary, as the link would have been automatically 'opened' on loading.

Control Block values to be set on entry:

BUFFER_
LENGTH: Set to 0x00.

Control Block values set on return:

RETURN_
CODE: 0x00 The action has been performed successfully.

0x06, 0x07, 0x08, 0x09, 0x0A
See Section "Notes on Return Codes" for further details.

HDLC_CLOSE (0x03)

This command causes the HDLC code to no longer send frames or to respond to incoming frames.

The HDLC link state is unchanged, but no transmissions will occur until an HDLC_OPEN command is executed.

Control block values to be set on entry:

BUFFER_
LENGTH: Set to 0x00.

Control block values set on return:

RETURN_
CODE: 0x00 The action has been performed successfully.

0x06, 0x07, 0x08, 0x09, 0x0A
See Section "Notes on Return Codes" for further details.

LINK_SETUP (0x04)

This command causes link initialization.

The HDLC code will attempt to enter the Asynchronous Balanced Mode (ABM) by positively responding to incoming SABM frames or issuing SABM frames of its own.

If 'auto_HDLC' has been enabled in the configuration file, then this command is not neccessary, as the station will automatically attempt to enter the ABM on loading.

Control block values to be set on entry:

BUFFER_
LENGTH: Set to 0x00.

Control block values on return:

RETURN_
CODE: 0x00 The action has been performed successfully.

0x03 The device is CLOSED and so no frames may be transmitted. Perform an HDLC_OPEN command.

0x06, 0x07, 0x08, 0x09, 0x0A
See Section "Notes on Return Codes" for further details.

LINK_DISCONNECT (0x05)

This command causes link disconnection by the issuing of a DISC frame. The user has the option of entering different HDLC states once this disconnection has been completed.

Control block values to be set on entry:

BUFFER_
LENGTH: Set to 0x01.

DATA: Offset 0x00 is used to define the HDLC state entered after the DISC frame has been issued, as follows:

0x00 - the link will not re-enter the ABM and incomming SABM frames will be responded to with a DM frame. The LINK_SETUP command must be issued to re-enter the ABM.

0x01 - the link will re-enter the ABM on receipt of a SABM from the remote station.

Control block values set on return:

RETURN_
CODE: 0x00 The action has been performed successfully.

0x02 The link is not currently in the ABM and so may not be disconnected.

0x03 The device is CLOSED and so no frames may be transmitted.

0x06, 0x07, 0x08, 0x09, 0x0A
See Section "Notes on Return Codes" for further details.

LINK_STATUS (0x06)

This command returns the HDLC status of the link, the number of Information frames queued for transmission and reception, the rotating Supervisory frame reception count and the station configuration.

Control Block values to be set on entry:

BUFFER-
LENGTH: Set to 0x00.

Control Block values set on return:

RETURN_
CODE:
- 0x00 The link is in the Disconnected mode.
- 0x01 The link is in the Asynchronous Balanced Mode (ABM), permitting data transfer.
- 0x06, 0x07, 0x08, 0x0A
  See Section "Notes on Return Codes" for further details.

BUFFER-
LENGTH: Set to 0x05 if a RETURN_CODE of 0x00 or 0x01 is received.

DATA (valid if a RETURN_CODE of 0x00 or 0x01 is received):

- Offset 0x00: the number of Information frames queued for transmission (0 to 7 frames).
- Offset 0x01: the number of incoming Information frames queued for reception by the application (0 to 7 frames).
- Offset 0x02: the station configuration as defined in the configuration file (or as set in the CONFIGURE_LINK command).
  - 0x01 configured as a DTE.
  - 0x02 configured as a DCE.
- Offset 0x03: reserved.
- Offset 0x04: the rotating Supervisory frame reception count. This count is incremented on the reception of every Supervisory frame and is useful in monitoring link activity. This value rotates between 0 and 255.

READ_HDLC_STATISTICS (0x07)

Retrieve HDLC level statistics on the operation of this station.

Control Block values to be set on entry:

BUFFER-
LENGTH: Set to 0x00.

Control Block values set on return:

RETURN_
CODE: 0x00 The action was performed successfully.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details.

BUFFER-
LENGTH: Set to 0x22 if a RETURN_CODE of 0x00 is received.

DATA (valid if a RETURN_CODE of 0x00 is received):

Note that each value listed below is a two byte unsigned short value and is set with the low byte first.

Offset
0x00,0x01: number of Information frames received and made available for reception by the application.

Offset
0x02,0x03: number of Information frames received out of sequence.

Offset
0x04, 0x05: number of Information frames received with no data field present.

Offset
0x06, 0x07: number of incoming frames discarded due to one of the following reasons:

The frame type was unsupported.

The frame was shorter than four bytes in length.

The frame was of an S or U format but had an illegal Information field attached.

Offset
0x08, 0x09: number of incoming frames whose data field exceeded the maximum configured data length.

Offset
0x0A, 0x0B: number of frames received with an invalid HDLC address.

Offset
0x0C, 0x0D: number of Information frames transmitted and acknowledged.

Offset
0x0E, 0x0F: number of Information

frames re-transmitted.

Offset
0x10, 0x11: number of T1 timeouts.

Offset
0x12, 0x13: number of SABM frames received.

Offset
0x14, 0x15: number of DISC frames received.

Offset
0x16, 0x17: number of DM frames received.

Offset
0x18, 0x19: number of FRMR frames received.

Offset
0x1A, 0x1B: number of SABM frames transmitted.

Offset
0x1C, 0x1D: number of DISC frames transmitted.

Offset
0x1E, 0x1F: number of DM frames transmitted.

Offset
0x20, 0x21: number of FRMR frames transmitted.

FLUSH_HDLC_STATISTICS (0x08)

The current values of the variables accessed by the READ_HDLC_STATISTICS command are reset to zero.

Control Block values to be set on entry:

BUFFER-
LENGTH: Set to 0x00.

Control Block values set on return:

RETURN_
CODE: 0x00 The action was performed successfully.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details.

READ_COMMS_ERR_STATS (0x09)

Retrieve the communications error statistics for the link.

Control Block values to be set on entry:

BUFFER-LENGTH: Set to 0x00.

Control Block values set on return:

RETURN_CODE:
- 0x00 The action was performed successfully.
- 0x06, 0x07, 0x08, 0x0A See Section "Notes on Return Codes" for further details.

BUFFER-LENGTH: Set to 0x0A if a RETURN_CODE of 0x00 is received.

DATA (valid if a RETURN_CODE of 0x00 is received):

Offset 0x00: number of receiver overrun errors.

Offset 0x01: number of receiver CRC errors.

Offset 0x02: number of abort frames received.

Offset 0x03: number of frames discarded at the interrupt level due to buffering constraints.

Offset 0x04: number of abort frames transmitted.

Offset 0x05: number of transmit underruns.

Offset 0x06: number of missed transmit underrun interrupts.

Offset 0x07: reserved for later use.

Offset 0x08: number of times DCD was found to be unexpectedly inactive.

Offset 0x09: number of times CTS was found to be unexpectedly inactive.

FLUSH_COMMS_ERR_STATS (0x0A)

The current values of the variables accessed by the READ_COMMS_ERR_STATS command are reset to zero.

Control Block values to be set on entry:

BUFFER-
LENGTH: Set to 0x00.

Control Block values set by XIP on return:

RETURN_
CODE: 0x00 The action has been performed successfully.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details .

SET_GLOBAL_VARIABLES (0x0B)

Set the status of DTR.

Control Block values to be set on entry:

BUFFER_
LENGTH: Set to 0x03.

DATA: Offset 0x00 reserved - set to 0x00.

Offset 0x01 DTR control as follows:
set to 0x01 to lower DTR.
set to 0x02 to raise DTR.

Offset 0x02 reserved - set to 0x00.

Control Block values set on return:

RETURN_
CODE: 0x00 The action has been performed successfully.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details.

READ_MODEM_STATUS (0x0C)

Read the current CTS and DCD status.

Control Block values to be set on entry:

BUFFER-
LENGTH: Set to 0x00.

Control Block values set on return:

RETURN_
CODE: 0x00 The action has been performed successfully.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details .

BUFFER-
LENGTH: Set to 0x01 if a RETURN_CODE of 0x00 is received.

DATA (valid if a RETURN_CODE of 0x00 is received):

Offset 0x00: The current CTS and DCD status as follows:

If bit 5 is set, then CTS is high.

If bit 3 is set, then DCD is high.

FLUSH_HDLC_DATA_BUFFERS (0x0D)

Flush the queued transmit and receive Information frame buffers.

Control Block values to be set on entry:

BUFFER_
LENGTH: Set to 0x00.

Control block values set on return:

RETURN_
CODE: 0x00 The action has been performed successfully.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details .

SEND_UNNUMBERED_INFORMATION_FRAME (0x10)

Send an Unnumbered Information frame to the card for onward transmission to the network.

Control Block values to be set on entry:

BUFFER_LENGTH: The length of the actual Unnumbered Information data field, plus two bytes. The maximum BUFFER_LENGTH is 1029 bytes.

PF_BIT: The status of the Poll/Final bit to be set in the UI frame.
If the P-bit is to be set, then this parameter should be set to 0x01, otherwise it should be set to 0x00.

DATA: The DATA area is formatted as follows:

Offset 0x00: the HDLC address to be used in this UI frame. This parameter is usually set to 0xFF.

Offset 0x01: reserved.

Offset 0x02 to 0xNN: the actual UI data field.

Control Block values set on return:

RETURN_CODE:

0x00 The UI frame has been queued for transmission.

0x01 The buffer used for UI frame formatting is already in use. Attempt to transmit the UI frame again after a short delay.

0x03 The link is currently CLOSED and an HDLC_OPEN command must be executed before attempting to transmit Information frames.

0x04 The length of the UI data field is longer than the maximum of 1027 bytes.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details .

HDLC_WRITE (0x11)

Send a data message (Information frame) to the card for onward transmission to the network.

Control Block values to be set on entry:

BUFFER_LENGTH: The length of the data message to be transmitted. The maximum data length is 1027 bytes and is dependant on the max_I_field_length defined in the configuration file.

PF_BIT: The status of the Poll/Final bit to be set in the HDLC header. In general, this parameter should be set to 0x00, but may be set to 0x01 if immediate confirmation of frame reception is required from the remote device.

DATA: The data to be transmitted.

Control Block values set on return:

RETURN_CODE:

0x00 The data has been queued for transmission.

0x01 The data was not queued due to the fact that the transmit window is closed.

The application is attempting to send data to the HDLC network at a rate faster than the network is able to receive the data. No data was sent to the S502 card and this same data should be re-sent in its entirety after a short delay.

0x02 The link is not currently in the ABM and data transfer is not possible.

0x03 The link is currently CLOSED and an HDLC_OPEN command must be executed before attempting to transmit Information frames.

0x04 The length of the data message to be transmitted exceeded the maximum data length max_I_field_length defined in the configuration file. The data was not transmitted.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details. .

READ_HDLC_CONFIGURATION (0x12)

Read the current HDLC configuration parameters.

Control Block values to be set on entry:

BUFFER_
LENGTH: Set to 0x00.

Control Block values set on return:

RETURN_
CODE: 0x00 The action was performed successfully.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details..

BUFFER_
LENGTH: Set to 0x0C if a RETURN_CODE of 0x00 is received.

DATA (valid if a RETURN_CODE of 0x00 is received):
The data is in the same format as that for the SET_HDLC_CONFIGURATION command.

SET_HDLC_CONFIGURATION (0x13)

Set the HDLC configuration parameters.

Note that no checking of the validity of the passed configuration data is performed by the HDLC code and the user must therefore ensure that this data is correct. Before using this command, it is recommended that the READ_HDLC_CONFIGURATION command be performed to check the accuracy of the configuration structure defined in the application code.

Control Block values to be set on entry:

BUFFER_
LENGTH: Set to 0x0C.

DATA: This area contains the configuration parameters. A detailed description of these parameters may be found in Section 3.

Offset	Size (bytes)	Parameter
0x00	1	A number corresponding to the baud rate generated by the SDLA card on pin 24 as follows:

Value	Baud Rate (bps)
0x00	External Clocking
0x01	1200
0x02	2400
0x03	4800
0x04	9600

0x05	19200
0x06	38400
0x07	45000
0x08	56000
0x09	64000
0x0A	74000
0x0B	112000
0x0C	128000

Offset	Size (bytes)	Parameter
0x01	1	The period timer T1.
0x02	1	The timer T2.
0x03	1	The N2 counter.
0x04	2	The maximum length of the HDLC Information frame data field. Valid values are between 1 and 1027 length.
0x06	1	The size of the frame level window (k).
0x07	1	The T4 parameter.
0x08	1	The 'auto_modem_err_fixup' parameter.
0x09	1	The 'auto_HDLC' parameter.
0x0A	1	The 'HDLC_config_options'.
0x0B	1	Defines the station's link level and packet level configuration and is set to 0x01 for a DTE and 0x00 for a DCE.

Control Block values set on return:

RETURN_CODE:

0x00 The action was performed successfully.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details. .

READ_CODE_VERSION (0x15)

Return the code versions for both the HDLC code and the XIP (if resident).

Control Block values to be set on entry:

BUFFER-
LENGTH: Set to 0x00.

Control Block values set on return:

RETURN_
CODE: 0x00 The action has been performed successfully.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for further details..

BUFFER-
LENGTH: Set to 0x09 if a RETURN_CODE of 0x00 is received.

DATA (valid if a RETURN_CODE of 0x00 is received):

The code versions are of the format:

main-version.sub-version

For example, XIP code version 2.01.

Offset 0x00 - 0x03: the version of the HDLC code running on the SDLA adapter.

Offset 0x04 - 0x07: the XIP code version (if resident).

Offset 0x08: reserved.

OPERATE_DATALINE_MONITOR (0x14)

Set the configuration of the dataline monitor or read the current monitor configuration.

Control Block values to be set on entry:

BUFFER-
LENGTH: Set to 0x02.

DATA:

Offset 0x00 miscellaneous monitor configuration bits as follows:

bit 0 if reset, then the dataline monitor is deactivated. If set, then the dataline monitor is activated.

bit 1 if set, then each traced frame will include a millisecond time stamp. Note that this time stamp rotates between 0 and 65535 milliseconds.

bit 2 if set, then the trace delay mode is activated. The trace delay mode prevents trace frames from being discarded due to limited on-board trace buffering. In effect, if the delay mode is enabled, then the actual transmission of HDLC frames is slowed down on the adapter to keep pace with the rate at which the application is reading trace frames from the board.

bit 3 if set, then HDLC I-frames will be traced.

bit 4 if set, then HDLC Supervisory frames will be traced.

bit 5 if set, then Unnumbered frames will be traced.

bit 6 if set, then all HDLC level frames will be traced.

bit 7 if set, then the current trace configuration will be returned to the application. Note that the trace configuration will not be set if this bit is enabled.

Offset 0x01 the trace deactivation timer. This timer is used in conjunction with the trace delay mode and automatically deactivates the trace in the case where the application does not read trace frames from the board at least once per defined deacitivation

period.

This value of this timer may be set from 1 to 8 seconds.

Control Block values set on return:

RETURN_CODE:
- 0x00 The action has been performed successfully.
- 0x06, 0x07, 0x08, 0x0A See Section "Notes on Return Codes" for further details..

BUFFER-LENGTH: Set to 0x01 if is the current trace configuration was requested.

DATA (valid if the current trace configuration was requested):
- Offset 0x00 miscellaneous monitor configuration bits as defined above.

READ_TRACE_DATA (0x16)

Read a trace frame from the SDLA adapter.

Control Block values to be set on entry:

BUFFER-LENGTH: Set to 0x00.

Control Block values set on return:

RETURN_CODE:
- 0x00 The action has been performed successfully and trace data for pick up in the DATA area of this interface structure.
- 0x01 No trace frames are currently available for the application.
- 0x02 The trace has not been activated.
- 0x06, 0x07, 0x08, 0x0A See Section "Notes on Return Codes" for further details..

BUFFER-LENGTH: Set to the length of the received trace structure if a RETURN_CODE of 0x00 is received.

DATA (valid if a RETURN_CODE of 0x00 is received):

The received trace structure in the following format:

Offset
0x00-0x01: the length of the actual trace data.

Offset 0x02: the type of trace is indicated in the low 2 bits as follows:

0x00 the trace is of a received frame.

0x01 the trace is of a transmitted frame.

0x02 the trace is of an HDLC error frame, in which case the trace type byte is decoded as follows:

0x12 the trace is of a received abort frame.

0x22 the trace is of a received frame with a CRC error.

0x32 the trace is of a received frame with an overrun error.

0x42 the trace is of a frame received of excessive length.

0x72 the trace is of an aborted transmitted frame (missed transmit interrupt).

0x82 the trace is of an aborted transmitted frame (missed transmit underrun interrupt).

Offset 0x03: the number of trace packets discarded since the last trace frame was passed to the application.

Offset
0x04-0x08: valid only if the XIP interface is being used, and is the time at which this transaction occurred (in binary coded decimal). The format of this four-byte time stamp is as follows:

Offset	Parameter
0x04	Month
0x05	Day
0x06	Hours
0x07	Minutes
0x08	Seconds

Offset
0x09-0x0A: the millisecond time stamp for the frame (valid only if time stamping has been activated). This time stamp rotates between 0 and 65535 milliseconds.

Offset
0x0B-0xNN: the actual contents of the frame. This includes all bytes in the frame, but excludes the opening and closing flags. For outgoing frames, the two CRC bytes are represented by the value 0xFFFF.

HDLC_READ (0x21)

Receive a data message (Information frame) from the network.

Control Block values to be set on entry:

BUFFER_
LENGTH: Set to 0x00.

Control block values set on return:

RETURN_
CODE: 0x00 Data has been received and is available for pick up in the DATA area of this control block.

0x01 No data is available at the station.

0x02 The link is not currently in the ABM and data transfer is not possible.

0x03 The link is currently CLOSED and an HDLC_OPEN command must be executed before attempting to receive Information frames.

0x06, 0x07, 0x08, 0x0A
See Section "Notes on Return Codes" for

BUFFER_
LENGTH: The length of the data message received.

(valid if a RETURN_CODE of 0x00 is received).

PF_BIT: The status of the HDLC Poll/Final bit in the received Information frame (valid if a RETURN_CODE of 0x00 is received). If this value is set to 0x00, the the P/F-bit was reset in the frame, otherwise the P/F-bit was set.

DATA (valid if a RETURN_CODE of 0x00 is received):
The actual I-frame data that has been received at the station.

7. Notes on Return Codes

There are return codes common to specific commands which require additional discussion. These return codes (hexadecimal values) are:

0x05 The command used is invalid.

0x06 An Unnumbered frame (SABM, DISC, DM, UA) was unexpectedly received while the link was in the Asynchronous Balanced Mode, or a UI frame has been received.

The BUFFER_LENGTH will be set to 0x01 (plus the length of the UI Data field if applicable) and the byte at offset 0x00 of the XIP structure DATA area indicates the type of Unnumbered frame received as follows:

0x01 - a SABM command frame was received.

0x02 - a DISC command frame was received.

0x03 - a DM response frame was received.

0x04 - a UI frame was received.

0x05 - a UA response frame was received.

If the received frame is a UI, then the DATA area is formatted as follows:

Offset 0x01: the HDLC address in the received UI frame.

Offset 0x02: reserved.

Offset 0x03 to 0xNN: The actual UI data field (of length BUFFER_LENGTH - 3).

If the received frame is a DISC or a DM, the Sangoma HDLC code will automatically attempt to re-enter the ABM by issuing SABMs.

Note: the command executed when this return code was passed to the application was not performed (i.e. any returned data is invalid) and this call should be repeated.

0x07 The link is in the Frame Reject mode.

The BUFFER_LENGTH will be set to 0x04 and the interface structure DATA area may be decoded as follows:

Offset 0x00: the source of the FRMR frame
0x01 - local
0x02 - remote

Offset
0x01-0x03: the actual FRMR Information field. The byte at offset 0x03 provides the reason for the FRMR as follows:

0x01 control field invalid or not implemented.

0x03 an S or U frame received with an illegal I-field attached.

0x04 an I-frame received whose data length exceeded the defined maximum.

0x08 an invalid Nr count.

The Sangoma HDLC level code will automatically attempt to reset the link when FRMR responses are received.

Note: the command executed when this return code was passed to the application was not performed (i.e. any returned data is invalid) and this call should be repeated.

0x08 A modem failure occurred - DCD and/or CTS were found to be unexpectedly low.

The cause of this failure is returned at offset 0x00 of the XIP data buffer and may be one of the following:

0x01 - DCD was found to be unexpectedly low.
0x02 - CTS was found to be unexpectedly low.

Executing a READ_COMMS_ERR_STATS will indicate the count of the various modem error types and the READ_MODEM_STATUS command will return the current status of DCD and CTS.

Note: the command executed when this return code was passed to the application was not performed (i.e. any returned data is invalid) and this call should be repeated.

0x09 The N2 retry limit has been exceeded, i.e. an unnumbered frame has been issued N2 times, but the remote device has still not responded.

The BUFFER_LENGTH will be set to 0x01 and the byte at offset 0x00 of the XIP structure DATA area indicates the type of Unnumbered frame concerned with this retry limit:

0x01 - a SABM command retry limit occurred.

0x02 - a DISC command retry limit occurred.

0x0A A SDLA card timeout occurred (valid if the XIP is resident).

The XIP call to the board was not processed in the specified time, indicating a hardware failure.

If this error code is returned, run XLOAD.EXE again, ensuring successful execution. On reoccurrence of this error, contact your Sangoma dealer.

8. PC/SDLA Interface Bytes

There are a number of bytes within the shared PC/SDLA memory area which may be useful for programmers using the shared memory interface. These bytes are:

The HDLC_RX_TX_STATUS_BYTE (offset 0x1EFF from the defined memory window base address). This byte is set as follows:

bits 0-3 set to the number of Information frames currently available for reception by the application.

bits 4-6 reserved.

bit 7 if this bit is set to 1, then there is space available for the queueing of at least one outgoing Information frame, i.e. an HDLC_WRITE command will not return an error code of 0x01 on completion.

The MISCELLANEOUS_HDLC_BITS (offset 0x1FF0 from the defined memory window base address). This byte is set as follows:

bit 0 set to 0 if the HDLC link is disconnected.
set to 1 if the HDLC link is in the asynchronous balanced mode (ABM).

bits 1-2 reserved.

bit 3 if this bit is set to 1, then there is trace data available for the application (valid if the line trace has been enabled).

bits 4-7 reserved.

9. General Programming Notes

Once completion of an interface command has occurred (the XIP returns to the calling application or the 'opp_flag' has been reset in the shared memory area), the application should examine the RETURN_CODE and other parameters relevant to the particular command. Note that the RETURN_CODE is most important, as it may indicate that the interface command was not successful. For example, if the command is an HDLC_WRITE and the RETURN_CODE is 0x01, then the Information frame was not sent to the adapter and this same HDLC_WRITE should be re-issued.

10. Example Code

Example code for interfacing with the XIP TSR

(WRITTEN FOR BORLAND C++, Version 4.0)

```
/*

To execute an XIP command, the following steps should be
performed:

       Set all the parameters in the XIP control block
       structure parameters required for the particular
       command. For example, if the command is an HDLC_WRITE,
       then the BUFFER_LENGTH, the PF_BIT and DATA areas must
       be completed.

       Set the BX register to the data segment and the DX
       register to the offset of the XIP control block
       structure.

       Call the XIP software interrupt.

       The XIP will carry out the defined command. It will then
       update the XIP control block at the application level
       with the required return code and, if applicable, the
       associated data buffer, data length etc.

       Once control is returned to your application, examine
       the RETURN_CODE and other parameters relevant to the
       particular XIP command.
*/

#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <ctype.h>
#include <mem.h>
#include <dos.h>

/* XIP commands */
#define LINK_STATUS    0x06     /* read the link status */
#define HDLC_WRITE     0x11     /* transmit data */
#define HDLC_READ      0x21     /* receive data */

#define XIP_INTERRUPT  0x6F     /* the XIP software interrupt */

/* the XIP calling structure */
typedef struct {
  char command;                /* XIP command */
  unsigned short buffer_length;/* buffer length */
```

```
  char return_code;            /* return code */
  char PF_bit;                 /* the setting of the
                              poll/final-bit */
  char reserved[11];           /* reserved for later use */
  char data[1040];             /* data area */
} XIP_CALL_STRUCT;
XIP_CALL_STRUCT XIP_struct;

int i;


void check_link_status(void);
void send_HDLC_I_frame(void);
void receive_HDLC_I_frame(void);
void execute_interface_command(void);
void exit_from_example(char, char);


void main()
{

   /*
   Assume that the 'auto_HDLC' configuration parameter is set
  to '1' in the HDLC configuration file, and so the
CONFIGURE_LINK and HDLC_OPEN commands are not required.
   */

   /* check to see that the link is in the ABM (asynchronous
   balanced mode) */
   check_link_status();

   /* transmit an HDLC I-frame */
   send_HDLC_I_frame();

   /* receive an HDLC I-frame */
   receive_HDLC_I_frame();

}



/*** check to see that the link is in the ABM (asynchronous
balanced mode) ***/

void check_link_status()
{
   /* loop until the link is in the ABM and Information frames
     may be transferred */
   do {
      /* set the command */
      XIP_struct.command = LINK_STATUS;

      /* set the length of the data buffer */
      XIP_struct.buffer_length = 0x00;
```



```
      /* perform the interface command */
      execute_interface_command();

   /* loop until the command is successful */
   } while(XIP_struct.return_code != LINK_IN_ABM);

   printf("\n\nThe link is in the ABM");
}



/*** send an HDLC Information frame ***/

void send_HDLC_I_frame()
{
   /* set the command */
   XIP_struct.command = HDLC_WRITE;

   /* reset the P-bit in the frame */
   XIP_struct.PF_bit = 0x00;

   /* set the actual data to be sent */
   sprintf(XIP_struct.data, "%s", "HDLC TEST FRAME");

   /* set the length of the data */
   XIP_struct.buffer_length = 15;

   /* perform the interface command */
   execute_interface_command();

   if(XIP_struct.return_code)
      printf("\n\nHDLC I-frame not sent (return code = %x)",
      XIP_struct.return_code);
   else
      printf("\n\nI-frame successfully sent");
}



/*** receive an HDLC Information frame ***/

void receive_HDLC_I_frame()
{
   /* set the command */
   XIP_struct.command = HDLC_READ;

   /* set the length of the data buffer */
   XIP_struct.buffer_length = 0x00;

   /* perform the interface command */
   execute_interface_command();
```

```
   /* if there is an I-frame available, then dispay the
received data */
   if(!XIP_struct.return_code) {
      /* display the received data */
      printf("\n\n%d bytes of data received (P-bit = %x)",
      XIP_struct.buffer_length, XIP_struct.PF_bit);
      printf("\nData: ");
      for(i = 0; i < XIP_struct.buffer_length; i ++)
         printf("%x ", XIP_struct.data[i]);
   }
   else
      printf("\n\nNo I-frame received");
}



/*** perform an interface command ***/

void execute_interface_command()
{
union REGS inregs;

   /* set the BX register to our Data Segment */
   inregs.x.bx = _DS;

   /* set the DX register to point to the XIP structure */
   inregs.x.dx = (unsigned short)&XIP_struct.command;

   /* call the XIP with a software interrupt */
   int86(XIP_INTERRUPT, &inregs, &inregs);
}



/*** exit to the operating system ***/

void exit_from_example(char command, char exit_code)
{
   printf("\n\nEXITING DUE TO RETURN CODE %x FOR COMMAND %x",
 exit_code, command);
   _exit((int)0x00);
}
```


Example showing interface to the shared memory area

(WRITTEN FOR BORLAND C++, Version 4.0)

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <ctype.h>
#include <mem.h>
#include <dos.h>

/* XIP commands */
#define LINK_STATUS   0x06   /* read the link status */
#define HDLC_WRITE    0x11   /* transmit data */
#define HDLC_READ     0x21   /* receive data */


/*
 Assume that the HDLC code was loaded with the configuration
file defining the memory address as 0xD000 and the memory
window as 0. The physical address of the SEND interface
mailbox is 0xD000:0x16B0 and the address of the RECEIVE
interface mailbox 0xD000:0x1AD0.
*/
#define SEND_MAILBOX_ADDRESS              0xD00016B0
#define RECEIVE_MAILBOX_ADDRESS           0xD0001AD0


/* for opp_flag use */
#define RESET  0x00
#define SET    0x01
#define ZERO   0x00


/* link status */
#define LINK_DISCONNECTED 0x00   /* the link is disconnected */
#define LINK_IN_ABM       0x01   /* the link is in the ABM */


/* the mailbox structure */
typedef struct {
  char opp_flag;                /* the opp flag */
  char command;                 /* XIP command */
  unsigned short buffer_length; /* buffer length */
  char return_code;             /* return code */
  char PF_bit;                  /* the setting of the
                                poll/final-bit */
  char reserved[10];            /* reserved for later use */
  char data[1040];              /* data area */
} MAILBOX_INTERFACE_STRUCT;
```

```
/* define far pointers to the SEND and RECEIVE interface
structures */
MAILBOX_INTERFACE_STRUCT _far *SEND_mailbox_ptr;
MAILBOX_INTERFACE_STRUCT _far *RECEIVE_mailbox_ptr;

int i;
char test_data[] = "HDLC TEST FRAME";
char utility_buffer[1040];

void check_link_status(void);
void send_HDLC_I_frame(void);
void receive_HDLC_I_frame(void);
void execute_interface_command(MAILBOX_INTERFACE_STRUCT
_far *);
void exit_from_example(char, char);


void main()
{
   /* set the SEND mailbox pointer to the physical address on
    the adapter */
   SEND_mailbox_ptr = (MAILBOX_INTERFACE_STRUCT
   _far*)SEND_MAILBOX_ADDRESS;

   /* set the RECEIVE mailbox pointer to the physical address
    on the adapter */
   RECEIVE_mailbox_ptr = (MAILBOX_INTERFACE_STRUCT
   _far*)RECEIVE_MAILBOX_ADDRESS;

   /*
   Assume that the 'auto_HDLC' configuration parameter is set
 to '1' in the HDLC configuration file, and so the
CONFIGURE_LINK and HDLC_OPEN commands are not required.
   */

   /* check to see that the link is in the ABM (asynchronous
   balanced mode) */
   check_link_status();

   /* transmit an HDLC I-frame */
   send_HDLC_I_frame();

   /* receive an HDLC I-frame */
   receive_HDLC_I_frame();

}
```



```
/*** check to see that the link is in the ABM (asynchronous
balanced mode) ***/

void check_link_status()
{
   /* loop until the link is in the ABM and Information frames
     may be transferred */
   do {
      /* set the command */
      SEND_mailbox_ptr->command = LINK_STATUS;

      /* set the length of the data buffer */
      SEND_mailbox_ptr->buffer_length = 0x00;

      /* perform the interface command */
      execute_interface_command(SEND_mailbox_ptr);

   /* loop until the command is successful */
   } while(SEND_mailbox_ptr->return_code != LINK_IN_ABM);

   printf("\n\nThe link is in the ABM");
}



/*** send an HDLC Information frame ***/

void send_HDLC_I_frame()
{
   /* set the command */
   SEND_mailbox_ptr->command = HDLC_WRITE;

   /* reset the P-bit in the frame */
   SEND_mailbox_ptr->PF_bit = RESET;

   /* set the actual data to be sent */
   _fmemcpy(SEND_mailbox_ptr->data, test_data, 15);

   /* set the length of the data */
   SEND_mailbox_ptr->buffer_length = 15;

   /* perform the interface command */
   execute_interface_command(SEND_mailbox_ptr);

   if(SEND_mailbox_ptr->return_code)
      printf("\n\nHDLC I-frame not sent (return code = %x)",
      SEND_mailbox_ptr->return_code);
   else
      printf("\n\nI-frame successfully sent");
}
```

```
/*** receive an HDLC Information frame ***/

void receive_HDLC_I_frame()
{
   /* set the command */
   RECEIVE_mailbox_ptr->command = HDLC_READ;

   /* set the length of the data buffer */
   RECEIVE_mailbox_ptr->buffer_length = 0x00;

   /* perform the interface command */
   execute_interface_command(RECEIVE_mailbox_ptr);

   /* if there is an I-frame available, then dispay the
received data */
   if(!RECEIVE_mailbox_ptr->return_code) {
      /* display the received data */
      printf("\n\n%d bytes of data received (P-bit = %x)",
      RECEIVE_mailbox_ptr->buffer_length,
RECEIVE_mailbox_ptr->PF_bit);
      printf("\nData: ");
      _fmemcpy(utility_buffer, RECEIVE_mailbox_ptr->data,
      RECEIVE_mailbox_ptr->buffer_length);
      for(i = 0; i < RECEIVE_mailbox_ptr->buffer_length; i ++)
         printf("%x ", utility_buffer[i]);
   }

   else
      printf("\n\nNo I-frame received");
}



/*** perform an interface command ***/

void execute_interface_command(mailbox_ptr)
MAILBOX_INTERFACE_STRUCT _far *mailbox_ptr;
{
long opp_flag_loops = 0x00;

   /* set the 'opp_flag' to initiate the processing of the
 command */
   mailbox_ptr->opp_flag = SET;

   /* Read the 'opp_flag' from the shared memory area and, when
      it has been reset, then the command has been completed.
    Note that the application should check to see that the
 'opp_flag' is reset within a defined time period after
being set. A suitable time out would be approximately one
      second (most commands would be completed with 1/100th of
     a second).
   */
   do {
      if((++ opp_flag_loops) == 200000) {
```



```
         printf("\nAdapter dead. Contact your Sangoma
representative.");
         _exit((int)ZERO);

      }
   /* loop until the 'opp_flag' has been reset */
   } while(mailbox_ptr->opp_flag != RESET);
}



/*** exit to the operating system ***/

void exit_from_example(char command, char exit_code)
{
   printf("\n\nEXITING DUE TO RETURN CODE %x FOR COMMAND %x",
 exit_code, command);
   _exit((int)0x00);
}
```

11. HDLC Implementation

HDLC Protocol Specifics

The implementation of the HDLC protocol on the S502 adapter corresponds as closely as possible to the specifications of the ISO 7776 document "Information processing systems - Data communication - High-level data link control procedures - Description of HDLC LAPB-compatible DTE data link procedures", 15 December 1986, with the following specifics:

Link Initialization

After issuing a CONFIGURE_LINK command and a subsequent HDLC_OPEN, the station issues DM responses at a period of T1 seconds. Also, all incoming frames with the P-bit set will be responded to with a DM frame.

Once the LINK_SETUP command is issued, the station will issue the DM frame N2 times and then send SABM commands. Incoming SABMs will be responded to with a UA, setting the link in the ABM.

Note that if auto_HDLC is enabled in the configuration file (as is usually done for an HDLC implementation), then the CONFIGURE_LINK, HDLC_OPEN and LINK_SETUP commands are automatically issued on code startup.

Link Disconnection

When the LINK_DISCONNECT command is used,

a DISC command is issued on the link at a period of T1 until a UA response is received. The station then issues DM responses at a period of T1 seconds.

Frame Reject Mode

The HDLC station will begin Frame Reject transmission under the following circumstances:

The Control field in the received frame does not allow an information field to be included with the frame, but an information field is present.

The function specified by the Control field has not been implemented.

An Information frame is received with an I-field which exceeds the maximum defined length.

When receiving a FRMR during the ABM, this station will attempt link re-initialization by issuing a SABM command.

12. Error messages

XLOAD.EXE

If XLOAD does not execute successfully, an error message will be displayed and an exit code will be returned. The error messages and corresponding exit codes (DOS ERRORLEVEL) are as follows:

"A command line error was found when executing XLOAD"
An invalid command line argument was used or the CODEFILE or CONFIG arguments were omitted (exit code of 1).

"The file FILENAME was not found"
A filename listed in the command line arguments was not found in the defined directory (exit code of 2).

"The parameter PARAMETER was not found in the configuration file"
There is an error in the HDLC configuration file and the listed PARAMETER could not be located (exit code of 3).

"The parameter PARAMETER read from the configuration file is invalid"
There is an error in the HDLC configuration file as the listed PARAMETER is invalid (exit code of 3).

"The code running on the adapter is not the same as the original downloaded code"
There is a memory or I/O port address conflict in your PC. Change the I/O port address and/or the memory

segment and memory window parameters (exit code of 4).

"The downloaded code is not running on the adapter"
The SDLA CPU has halted. Contact your Sangoma representative (exit code of 5).

"The S502 adapter is configured with the incorrect serial communications cXIP"
Contact your Sangoma representative (exit code of 6).

XIP.COM

If XIP does not execute successfully (exit code of 0x00), an error message will be displayed and a non-zero exit code will be returned. The error messages and corresponding exit codes are as follows:

"THE XIP TSR IS ALREADY RESIDENT"
Multiple copies of this TSR may not be run at the same software interrupt address (exit code of 0x01).

"A COMMAND LINE ERROR WAS FOUND"
An invalid command line argument was used and the valid arguments are displayed (exit code of 0x02).

"THE DEFINED HDLC/HDLC CONFIGURATION FILE HAS NOT BEEN FOUND"
The configuration filename listed in the command line arguments (or the default configuration file) was not found in the defined directory (exit code of 0x03).

"ERROR IN READING THE HDLC/HDLC CONFIGURATION FILE"
The configuration file could not be opened for reading (exit code of 0x04).

"THE HDLC CONFIGURATION FILE IS OF EXCESSIVE LENGTH"
The configuration file could not be read into the allocated buffer space (exit code of 0x05).

"THE I/O PORT ADDRESS OR THE MEMORY CONFIGURATION PARAMETERS WERE NOT FOUND IN THE CONFIGURATION FILE"
Check the defined configuration file

for the presence of these two parameters (exit code of 0x06).

X25_TST.EXE

"A COMMAND LINE ERROR WAS FOUND"
An invalid command line argument was used (exit code of 1).

Index