NPI/SNA

User's Guide

Document Version 1.0

September 2000

Copyright © GCOM, Inc.
All rights reserved.

© 1998 GCOM, Inc. All rights reserved.

Non-proprietary—Provided that this notice of copyright is included, this document may be copied in its entirety without alteration. Permission to publish excerpts should be obtained from GCOM, Inc.

GCOM reserves the right to revise this publication and to make changes in content without obligation on the part of GCOM to provide notification of such revision or change. The information in this document is believed to be accurate and complete on the date printed on the title page. No responsibility is assumed for errors that may exists in this document.

Rsystem is a registered trademark of GCOM, Inc. UNIX is a registered trademark of UNIX Systems Laboratories, Inc. in the U.S. and other countries. SCO is a trademark of the Santa Cruz Operation, Inc. IBM PC, IBM PC/AT, OS/2 and PC DOS are registered trademarks of International Business Machines Corporation. All other brand product names mentioned herein are the trademarks or registered trademarks of their respective owners.

Any provision of this product and its manual to the U.S Government is with “Restricted Rights”: Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at 252.227-7013 of the DoD FAR Supplement.

This manual was written, formatted and produced by technical writer Geoff Gerrietts using Vim 5, FrameMaker 5, and CorelDraw 7 on a Microsoft Windows platform with the help of subject matter specialists Dave Grothe and Carlton Mills.

This manual was printed in the U.S.A.

FOR FURTHER INFORMATION

If you want more information about GCOM products, contact us at:

GCOM, Inc.
1800 Woodfield
Savoy, IL 61874
(217) 351-4241
FAX: (217) 351-4240
e-mail: support@gcom.com
homepage: http://gcom.com

CONTENTS

SECTION 1 — ***7*** *Using GCOM's NPI/SNA*

9 Components of an SNA Stack
11 Starting SNA
11 *The Start Script*
11 *The Sample Script*
13 Getting SNA Configured
13 *Layout of the Configs Files*
15 *Unlisted Parameters*
15 *Multiple Occurances of the Same Parameter*

SECTION 2 — ***17*** *Configuring NPI/SNA*

19 Data Sizes
21 Data Window Sizes
23 Setting Addresses
25 Connection Style
27 Configuring Relationships
27 *Frame-Level DTE/DCE*
27 *SDLC Primary/Secondary*
27 *Full or Half Duplex Line*
29 Special SNA Scenarios
31 *Synchronous Communications Clocks*

SECTION 3 — ***35*** *Using NPI/SNA API*

37 The NPI/SNA API Model
38 Summary of Routines
38 *Initializing/Shutting Down*
38 *Connecting*
39 *Data Transfer*
39 *Protocol Features*
40 *Utility Routines*

1

Using GCOM's NPI/SNA

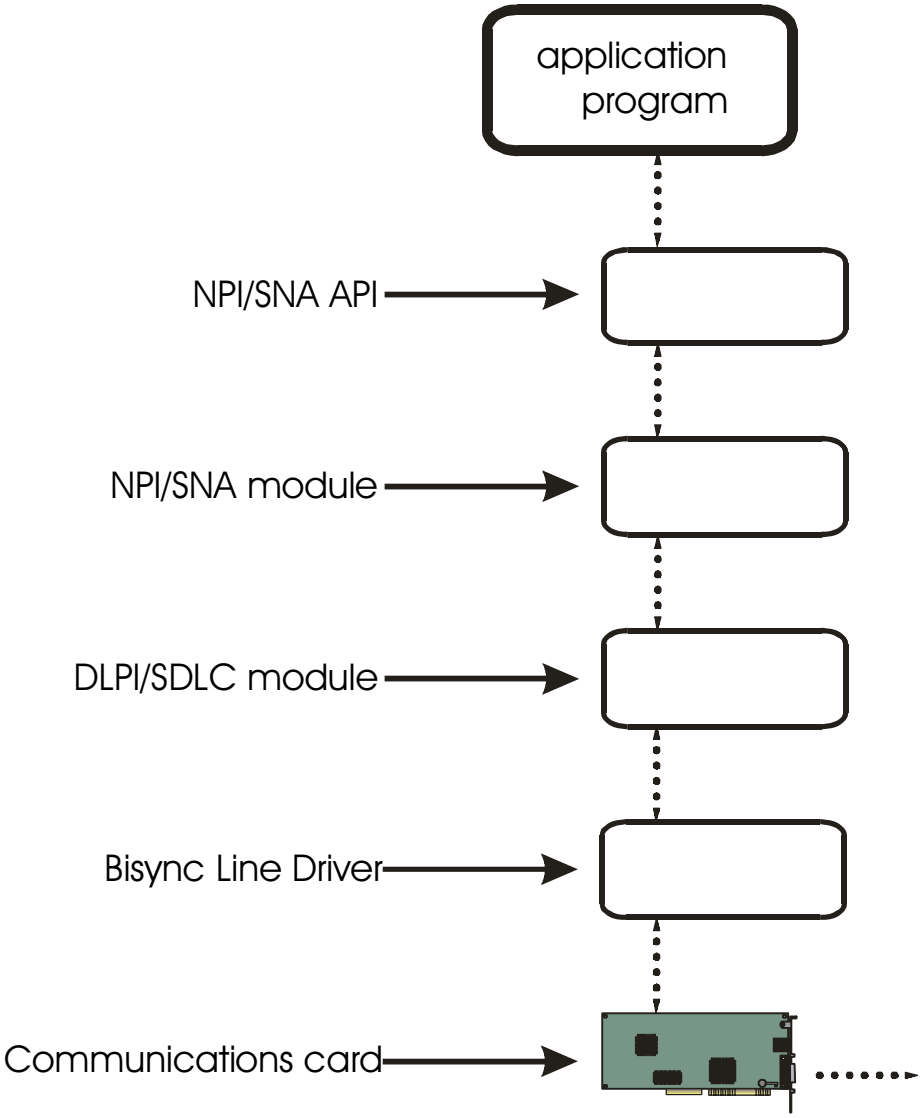


Figure 1 Modules within an SNA Stack

Components of an SNA Stack

A GCOM STREAMS SNA protocol stack has four primary components: the line driver, the STREAMS SDLC module, the STREAMS SNA module, and the NPI/SNA API.

The line driver handles manipulation of the actual physical device, managing modem signals, on-board buffers, and other issues specific to the hardware.

The second component, the STREAMS SDLC module, handles responding to SDLC polls, handles other frame-level events, and maintains a reliable point-to-point connection.

The third component, the STREAMS SNA module, handles the actual SNA events, managing network routing and end-to-end delivery.

The final component, the NPI/SNA API provides application programs access to the facilities of the protocol modules and the line driver.

An application program will use the NPI/SNA API to provide communications services to an application, and the protocol stack itself will use a synchronous communications card to send data across a synchronous serial line.

```
Gcom_dlpi -X
Gcom_monitor -B -N -E -U1 -q -e./configs.sna.example
```

Figure 2 The sample start script, **start.sna.example**

Starting SNA

The Start Script

Gcom provides scripts to start the Gcom protocol stack for most common configurations. These scripts are intended to be executed from the directory they reside in, and would ordinarily be called from a system's startup scripts.

Several variants are available for SNA and should be present in **/usr/lib/gcom/sna/**. This User's Guide will use the file **start.sna.example** for sample discussions.

The Sample Script

A copy of the sample file is at left. The script is only two lines long.

```
Gcom_dlpi -X
```

This line cleans up any previously running copies of Gcom's protocol software.

```
Gcom_monitor -B -N -E -U1 -q -e./configs.sna.example
```

This line starts the Gcom software. The options have the following meanings:

-B Run *Gcom_monitor* in the background.

-N Configure NPI (the SNA protocol) as well as DLPI (the SDLC protocol)

-E Honor user requests to terminate sent by *Gcom_dlpi -X* or *Gcom_dlpi -E*

-U Number of CDI UPAs (lines) to configure. Our sample start script instructs *Gcom_monitor* to configure a single line.

-q Quiet: minimize console output.

-e Selects the file to parse for configuration information.

```
cdi.mod
    n_minors    = 2
    n_upas      = 2
    n_lpas      = 00
dlpi.mod
    n_minors    = 15
    n_upas      = 15
    n_lpas      = 5
npi.mod
    n_minors    = 200
    n_upas      = 200
    n_lpas      = 2
rsys.mod
    bfrsize         = 950       # max RU size + 9 + 100
    r_unix_ticks    = 5
    r_timeout_ms    = 50
cdip.*
    ix25mode        = 01        # DMA
    ix25mode        = 00        # non-DMA
    hertz           = 9830400
    ix25_clks       = 00        # TxC, RxC both inputs
    baudrate        = 38400
    ix25_chip_type  = 13        # HD64570
    ix25_optns      = 0x0a      # free run, ignore rcv aborts
    mdm_msk         = 3
    maxfrmsz        = 0         # default to rsys bfrsize
cdiu.*
    poll_optns      = 0
    pollhsz         = 110       # poll trace table size
    poll_lap_type   = 4         # nrm_sec
    poll_delay      = 50        # milliseconds
dlpip.*
    n2              = 40        # for NRM sec
    fdte            = 1         # DTE
   maxfsize       = 460      # max # of bytes in the info part of I-frame
    frmwsize        = 7         # Common window size
    dl_off_delay    = 6
    lap_type        = 4         # NRM Secondary, two way alternate
    frm_optns3      = 0x0018    # SDLC protocol, interpret XIDs
    fhistsz         = 168       # large trace table
dlpip.1
    prim_addr       = 1, 0xFF   # respond to XID on addr FF
    sec_addr        = 1, 0x01
dlpiu.1
    dl_lwr_module_type      = 10            # SNA
    dl_dlpi_provider_lpa    = -1            # to attach
    dl_dlsap                = 1             # bind address (1, 0x01)
    dl_options              = 0xff00        # overkill on cause/diag
    sp_histsz               = 90            # trace table size
    sp_xid_node_id          = 0x04500001
    sp_options              = 0             # no options is the default
    sp_term_options         = 0x20          # NOTIFY, NO UNBIND
    sp_ext_options          = 0x00
    sp_sess_histsz          = 30
```

Figure 3 The sample configuration file **configs.sna.example**

Getting SNA Configured

Setting up an SNA installation requires setting up a **configs** file and a **start** script to run it. GCOM provides sample **configs** files and **start** scripts in the **/usr/lib/gcom/sna** directory. The files **config.example** and **start.example** are used as examples throughout this manual.

Layout of the Configs Files

The file splits into multiple sections, each headed by a flush-left section name. Names consist of a stem followed by an extension. The stem is the section category, while the extension is either '`.*`' indicating that the contents of the section applies to all sections in the category, or '`.mod`', which is used only in setting up the baseline resources for STREAMS, or a dot followed by a number, like '`.1`' or '`.2`'. This latter form configures a specific instance of the section category.

The first four sections configure the STREAMS facility directly, reserving adequate resources for the communications which the system will need to handle. These sections all conclude with a '`.mod`' extension.

Sections beginning with the `cdip` stem will come next. These sections configure the line driver. In **configs.example**, one section, `cdip.1`, configures a specific instance of the line driver.

The `cdiu` sections and `dlpip` sections configure the STREAMS SDLC driver and the polling controller module. In **configs.example**, `cdiu.*` handles all polling controllers, and `dlpip.*` configures all SDLC stations. The `dlpip.1` section configures one specific SDLC station, the only station configured in this example.

The `dlpiu` section configures the STREAMS SNA Server. In **configs.example**, `dlpiu.1` configures SNA.

```
cdi.mod
    n_minors    = 2
    n_upas      = 2
    n_lpas      = 00
dlpi.mod
    n_minors    = 15
    n_upas      = 15
    n_lpas      = 5
npi.mod
    n_minors    = 200
    n_upas      = 200
    n_lpas      = 2
rsys.mod
    bfrsize         = 950       # max RU size + 9 + 100
    r_unix_ticks    = 5
    r_timeout_ms    = 50
cdip.*
    ix25mode        = 01        # DMA
    ix25mode        = 00        # non-DMA
    hertz           = 9830400
    ix25_clks       = 00        # TxC, RxC both inputs
    baudrate        = 38400
    ix25_chip_type  = 13        # HD64570
    ix25_optns      = 0x0a      # free run, ignore rcv aborts
    mdm_msk         = 3
    maxfrmsz        = 0         # default to rsys bfrsize
cdiu.*
    poll_optns      = 0
    pollhsz         = 110       # poll trace table size
    poll_lap_type   = 4         # nrm_sec
    poll_delay      = 50        # milliseconds
dlpip.*
    n2              = 40        # for NRM sec
    fdte            = 1         # DTE
   maxfsize       = 460      # max # of bytes in the info part of I-frame
    frmwsize        = 7         # Common window size
    dl_off_delay    = 6
    lap_type        = 4         # NRM Secondary, two way alternate
    frm_optns3      = 0x0018    # SDLC protocol, interpret XIDs
    fhistsz         = 168       # large trace table
dlpip.1
    prim_addr       = 1, 0xFF   # respond to XID on addr FF
    sec_addr        = 1, 0x01
dlpiu.1
    dl_lwr_module_type      = 10            # SNA
    dl_dlpi_provider_lpa    = -1            # to attach
    dl_dlsap                = 1             # bind address (1, 0x01)
    dl_options              = 0xff00        # overkill on cause/diag
    sp_histsz               = 90            # trace table size
    sp_xid_node_id          = 0x04500001
    sp_options              = 0             # no options is the default
    sp_term_options         = 0x20          # NOTIFY, NO UNBIND
    sp_ext_options          = 0x00
    sp_sess_histsz          = 30
```

Figure 4 The sample configuration file **configs.sna.example**

Unlisted Parameters

Some configuration files will have parameter/value pairs in addition to the ones shown at left. Information on these additional parameters can be found in the *STREAMS Administrator's Guide*, should modification be required. Generally, these parameters will not require user modification.

Multiple Occurances of the Same Parameter

Some files may have multiple entries for the same parameter, like the sample file at left has for *ix25mode*. In such cases, the last value is significant, overriding the previous settings. This mechanism provides a useful way to maintain multiple settings in a file.

2

Configuring NPI/SNA

```
cdi.mod
    n_minors    = 2
    n_upas      = 2
    n_lpas      = 00
dlpi.mod
    n_minors    = 15
    n_upas      = 15
    n_lpas      = 5
npi.mod
    n_minors    = 200
    n_upas      = 200
    n_lpas      = 2
rsys.mod
    bfrsize         = 950       # max RU size + 9 + 100
    r_unix_ticks    = 5
    r_timeout_ms    = 50
cdip.*
    ix25mode        = 01        # DMA
    ix25mode        = 00        # non-DMA
    hertz           = 9830400
    ix25_clks       = 00        # TxC, RxC both inputs
    baudrate        = 38400
    ix25_chip_type  = 13        # HD64570
    ix25_optns      = 0x0a      # free run, ignore rcv aborts
    mdm_msk         = 3
    maxfrmsz        = 0         # default to rsys bfrsize
cdiu.*
    poll_optns      = 0
    pollhsz         = 110       # poll trace table size
    poll_lap_type   = 4         # nrm_sec
    poll_delay      = 50        # milliseconds
dlpip.*
    n2              = 40        # for NRM sec
    fdte            = 1         # DTE
   maxfsize       = 460      # max # of bytes in the info part of I-frame
    frmwsize        = 7         # Common window size
    dl_off_delay    = 6
    lap_type        = 4         # NRM Secondary, two way alternate
    frm_optns3      = 0x0018    # SDLC protocol, interpret XIDs
    fhistsz         = 168       # large trace table
dlpip.1
    prim_addr       = 1, 0xFF   # respond to XID on addr FF
    sec_addr        = 1, 0x01
dlpiu.1
    dl_lwr_module_type      = 10            # SNA
    dl_dlpi_provider_lpa    = -1            # to attach
    dl_dlsap                = 1             # bind address (1, 0x01)
    dl_options              = 0xff00        # overkill on cause/diag
    sp_histsz               = 90            # trace table size
    sp_xid_node_id          = 0x04500001
    sp_options              = 0             # no options is the default
    sp_term_options         = 0x20          # NOTIFY, NO UNBIND
    sp_ext_options          = 0x00
    sp_sess_histsz          = 30
```

Figure 5 The sample configuration file **configs.sna.example**

Data Sizes

An SNA protocol stack views buffers differently at each layer of the protocol. At each layer, the size of these buffers can be configured, allowing optimal flexibility.

maxfsize

In the `dlpip.`*n* sections. Defines the maximum amount of data that can be placed into the "information portion" of an I-frame. This information is used to define the size of a segment, if sementation is allowed for a particular session. The value of this parameter depends on the configuration of the front end device to which the SNA controller is attached. This information should be obtained from your system administrator.

maxfrmsz

In the `cdip.`*n* sections. Maximum frame size for the line driver. Includes space for a packet header and frame header. *maxfsize* + 4 should prove adequate, but usually, setting this parameter to 0 is the safest option. This will cause the monitor to calculate the appropriate value from the size of the Rsystem buffers.

bfrsize

In `rsys.mod`. Overall buffer size. Must be at least as large as *maxfrmsz*. We use a generously large *maxfsize* + 390 in the example, but a more general guideline would base the size off the largest RU which will be received. The system's administrator should be able to provide a figure for maximum RU size. Adding 120 to the maximum RU size should allow adequate working room.

```
cdi.mod
    n_minors    = 2
    n_upas      = 2
    n_lpas      = 00
dlpi.mod
    n_minors    = 15
    n_upas      = 15
    n_lpas      = 5
npi.mod
    n_minors    = 200
    n_upas      = 200
    n_lpas      = 2
rsys.mod
    bfrsize         = 950       # max RU size + 9 + 100
    r_unix_ticks    = 5
    r_timeout_ms    = 50
cdip.*
    ix25mode        = 01        # DMA
    ix25mode        = 00        # non-DMA
    hertz           = 9830400
    ix25_clks       = 00        # TxC, RxC both inputs
    baudrate        = 38400
    ix25_chip_type  = 13        # HD64570
    ix25_optns      = 0x0a      # free run, ignore rcv aborts
    mdm_msk         = 3
    maxfrmsz        = 0         # default to rsys bfrsize
cdiu.*
    poll_optns      = 0
    pollhsz         = 110       # poll trace table size
    poll_lap_type   = 4         # nrm_sec
    poll_delay      = 50        # milliseconds
dlpip.*
    n2              = 40        # for NRM sec
    fdte            = 1         # DTE
   maxfsize       = 460      # max # of bytes in the info part of I-frame
    frmwsize        = 7         # Common window size
    dl_off_delay    = 6
    lap_type        = 4         # NRM Secondary, two way alternate
    frm_optns3      = 0x0018    # SDLC protocol, interpret XIDs
    fhistsz         = 168       # large trace table
dlpip.1
    prim_addr       = 1, 0xFF   # respond to XID on addr FF
    sec_addr        = 1, 0x01
dlpiu.1
    dl_lwr_module_type      = 10            # SNA
    dl_dlpi_provider_lpa    = -1            # to attach
    dl_dlsap                = 1             # bind address (1, 0x01)
    dl_options              = 0xff00        # overkill on cause/diag
    sp_histsz               = 90            # trace table size
    sp_xid_node_id          = 0x04500001
    sp_options              = 0             # no options is the default
    sp_term_options         = 0x20          # NOTIFY, NO UNBIND
    sp_ext_options          = 0x00
    sp_sess_histsz          = 30
```

Figure 6 The sample configuration file **configs.sna.example**

Data Window Sizes

In conjunction with data sizes, the data window sizes determine how much data can be transmitted before requiring an acknowledgement from the receiving system. If using normal sequence numbers, these values must be between 1 and 7 and must match the target system's configuration. If using extended sequence numbers, these values may be between 1 and 127 and must match the target system's configuration.

frmwsize

Number of frames in the frame-level (SDLC) window. Must match the value used by the system at the other end of the physical line. Configured in the `dlpip.`*n* sections.

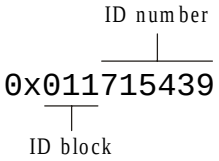
```
cdi.mod
    n_minors    = 2
    n_upas      = 2
    n_lpas      = 00
dlpi.mod
    n_minors    = 15
    n_upas      = 15
    n_lpas      = 5
npi.mod
    n_minors    = 200
    n_upas      = 200
    n_lpas      = 2
rsys.mod
    bfrsize         = 950       # max RU size + 9 + 100
    r_unix_ticks    = 5
    r_timeout_ms    = 50
cdip.*
    ix25mode        = 01        # DMA
    ix25mode        = 00        # non-DMA
    hertz           = 9830400
    ix25_clks       = 00        # TxC, RxC both inputs
    baudrate        = 38400
    ix25_chip_type  = 13        # HD64570
    ix25_optns      = 0x0a      # free run, ignore rcv aborts
    mdm_msk         = 3
    maxfrmsz        = 0         # default to rsys bfrsize
cdiu.*
    poll_optns      = 0
    pollhsz         = 110       # poll trace table size
    poll_lap_type   = 4         # nrm_sec
    poll_delay      = 50        # milliseconds
dlpip.*
    n2              = 40        # for NRM sec
    fdte            = 1         # DTE
   maxfsize       = 460      # max # of bytes in the info part of I-frame
    frmwsize        = 7         # Common window size
    dl_off_delay    = 6
    lap_type        = 4         # NRM Secondary, two way alternate
    frm_optns3      = 0x0018    # SDLC protocol, interpret XIDs
    fhistsz         = 168       # large trace table
dlpip.1
    prim_addr       = 1, 0xFF   # respond to XID on addr FF
    sec_addr        = 1, 0x01
dlpiu.1
    dl_lwr_module_type      = 10            # SNA
    dl_dlpi_provider_lpa    = -1            # to attach
    dl_dlsap                = 1             # bind address (1, 0x01)
    dl_options              = 0xff00        # overkill on cause/diag
    sp_histsz               = 90            # trace table size
    sp_xid_node_id          = 0x04500001
    sp_options              = 0             # no options is the default
    sp_term_options         = 0x20          # NOTIFY, NO UNBIND
    sp_ext_options          = 0x00
    sp_sess_histsz          = 30
```

Figure 7 The sample configuration file **configs.sna.example**

Setting Addresses

sp_xid_node_id

The *sp_xid_node_id* is the node ID the SNA host will use to identify the SNA station. This is a 32-bit number with the following general format:



sp_options 0x02 (force XID format 1)

If this bit is set, a format 1 XID will be sent instead of a format 0 XID. The extra bytes in the format 0 XID will be filled in automatically by the SNA Server.

dl_dlsap

The Service Access Point (SAP) to use for the link layer. This should correspond to the address of the HDLC station.

prim_addr

This may be set for each SDLC station, in the **dlpip.*** section or in each of the **dlpip.*n*** sections. It must be set to 1, 0xFF if the primary station wil send XID commands using the general poll address (0xFF). Otherwise, this parameter can be ignored.

sec_addr

This is set for each SDLC station, in the **dlpip.*n*** sections. This is the station's address.

```
cdi.mod
    n_minors    = 2
    n_upas      = 2
    n_lpas      = 00
dlpi.mod
    n_minors    = 15
    n_upas      = 15
    n_lpas      = 5
npi.mod
    n_minors    = 200
    n_upas      = 200
    n_lpas      = 2
rsys.mod
    bfrsize         = 950       # max RU size + 9 + 100
    r_unix_ticks    = 5
    r_timeout_ms    = 50
cdip.*
    ix25mode        = 01        # DMA
    ix25mode        = 00        # non-DMA
    hertz           = 9830400
    ix25_clks       = 00        # TxC, RxC both inputs
    baudrate        = 38400
    ix25_chip_type  = 13        # HD64570
    ix25_optns      = 0x0a      # free run, ignore rcv aborts
    mdm_msk         = 3
    maxfrmsz        = 0         # default to rsys bfrsize
cdiu.*
    poll_optns      = 0
    pollhsz         = 110       # poll trace table size
    poll_lap_type   = 4         # nrm_sec
    poll_delay      = 50        # milliseconds
dlpip.*
    n2              = 40        # for NRM sec
    fdte            = 1         # DTE
   maxfsize       = 460      # max # of bytes in the info part of I-frame
    frmwsize        = 7         # Common window size
    dl_off_delay    = 6
    lap_type        = 4         # NRM Secondary, two way alternate
    frm_optns3      = 0x0018    # SDLC protocol, interpret XIDs
    fhistsz         = 168       # large trace table
dlpip.1
    prim_addr       = 1, 0xFF   # respond to XID on addr FF
    sec_addr        = 1, 0x01
dlpiu.1
    dl_lwr_module_type      = 10            # SNA
    dl_dlpi_provider_lpa    = -1            # to attach
    dl_dlsap                = 1             # bind address (1, 0x01)
    dl_options              = 0xff00        # overkill on cause/diag
    sp_histsz               = 90            # trace table size
    sp_xid_node_id          = 0x04500001
    sp_options              = 0             # no options is the default
    sp_term_options         = 0x20          # NOTIFY, NO UNBIND
    sp_ext_options          = 0x00
    sp_sess_histsz          = 30
```

Figure 8 The sample configuration file **configs.sna.example**

Connection Style

Several of the *sp_ext_options* manage the way a connection is set up. These options provide for variant connection mechanisms.

sp_ext_options 0x20 (No LU Pool)

This option prevents use of a wildcard when requesting a connection. It also allows an application to request an LU prior to the LU being activated. The parameter *sp_connect_time* will then determine how long the connection request may be held before an ACTLU is received for the LU.

sp_ext_options 0x40 (Wait for LU Connect)

If the No LU Pool flag is set (*sp_ext_options* 0x20), this will prevent the link from being established until at least one LU requests a connection. This will be accomplished by simply not responding to the SNRM; if the connection is a dialup connection, the LU Send Disconnect bit should be set to allow the connection to be disconnected.

sp_ext_options 0x80 (LU Send Disconnect)

If this option is set in conjunction with the No LU Pool (*sp_ext_options* 0x20) and Wait for LU Connect (*sp_ext_options* 0x40) options, the SNA Server will respond to an SNRM with a Disconnect until an LU has requested a connection.

sp_connect_time

This parameter sets the amount of time a connection request can be held before an ACTLU request is received for the LU. Must be set in conjuction with the No LU Pool (*sp_ext_options* 0x20) option. This value is in seconds.

```
cdi.mod
    n_minors    = 2
    n_upas      = 2
    n_lpas      = 00
dlpi.mod
    n_minors    = 15
    n_upas      = 15
    n_lpas      = 5
npi.mod
    n_minors    = 200
    n_upas      = 200
    n_lpas      = 2
rsys.mod
    bfrsize         = 950       # max RU size + 9 + 100
    r_unix_ticks    = 5
    r_timeout_ms    = 50
cdip.*
    ix25mode        = 01        # DMA
    ix25mode        = 00        # non-DMA
    hertz           = 9830400
    ix25_clks       = 00        # TxC, RxC both inputs
    baudrate        = 38400
    ix25_chip_type  = 13        # HD64570
    ix25_optns      = 0x0a      # free run, ignore rcv aborts
    mdm_msk         = 3
    maxfrmsz        = 0         # default to rsys bfrsize
cdiu.*
    poll_optns      = 0
    pollhsz         = 110       # poll trace table size
    poll_lap_type   = 4         # nrm_sec
    poll_delay      = 50        # milliseconds
dlpip.*
    n2              = 40        # for NRM sec
    fdte            = 1         # DTE
    maxfsize       = 460      # max # of bytes in the info part of I-frame
    frmwsize        = 7         # Common window size
    dl_off_delay    = 6
    lap_type        = 4         # NRM Secondary, two way alternate
    frm_optns3      = 0x0018    # SDLC protocol, interpret XIDs
    fhistsz         = 168       # large trace table
dlpip.1
    prim_addr       = 1, 0xFF   # respond to XID on addr FF
    sec_addr        = 1, 0x01
dlpiu.1
    dl_lwr_module_type      = 10            # SNA
    dl_dlpi_provider_lpa    = -1            # to attach
    dl_dlsap                = 1             # bind address (1, 0x01)
    dl_options              = 0xff00        # overkill on cause/diag
    sp_histsz               = 90            # trace table size
    sp_xid_node_id          = 0x04500001
    sp_options              = 0             # no options is the default
    sp_term_options         = 0x20          # NOTIFY, NO UNBIND
    sp_ext_options          = 0x00
    sp_sess_histsz          = 30
```

Figure 9 The sample configuration file **configs.sna.example**

Configuring Relationships

The configuration file defines several relationships between the SNA Server and other SNA hosts.

Frame-Level DTE/DCE

fdte

This parameter is in the **dlpip.*** sections. If SDLC should be acting as DCE, this parameter should be set to 1. If it should be acting as DTE, this parameter should be 0.

SDLC Primary/Secondary

lap_type

The *lap_type* parameter is set in the **dlpip.*** sections. It specifies whether the SDLC station should be a primary or secondary by selecting one of the two NRM modes:

3 *Normal Response Mode primary, Two-Way Alternate.* This mode sets the SDLC station up to use a half-duplex procedure as the SDLC primary. This can be used over a half-duplex or full-duplex line. This is the LAP type used for mult-drop configurations or when acting as an IBM front end processor.

4 *Normal Response Mode secondary, Two-Way Alternate.* This mode sets the SDLC station up to use a half-duplex procedure as the SDLC secondary. This can be used over a half-duplex or full-duplex line. This is the LAP type used for IBM cluster controllers.

Full or Half Duplex Line

frm_optns2 0x80 (half-duplex)

If the SDLC line should be treated as a half-duplex line, the 0x80 bit in *frm_optns2* needs to be turned on. This will produce a value 0xC0 for half-duplex lines when combined with the standard 0x40. A full duplex line will usually use only 0x40 for *frm_optns2*.

```
cdi.mod
    n_minors    = 2
    n_upas      = 2
    n_lpas      = 00
dlpi.mod
    n_minors    = 15
    n_upas      = 15
    n_lpas      = 5
npi.mod
    n_minors    = 200
    n_upas      = 200
    n_lpas      = 2
rsys.mod
    bfrsize         = 950       # max RU size + 9 + 100
    r_unix_ticks    = 5
    r_timeout_ms    = 50
cdip.*
    ix25mode        = 01        # DMA
    ix25mode        = 00        # non-DMA
    hertz           = 9830400
    ix25_clks       = 00        # TxC, RxC both inputs
    baudrate        = 38400
    ix25_chip_type  = 13        # HD64570
    ix25_optns      = 0x0a      # free run, ignore rcv aborts
    mdm_msk         = 3
    maxfrmsz        = 0         # default to rsys bfrsize
cdiu.*
    poll_optns      = 0
    pollhsz         = 110       # poll trace table size
    poll_lap_type   = 4         # nrm_sec
    poll_delay      = 50        # milliseconds
dlpip.*
    n2              = 40        # for NRM sec
    fdte            = 1         # DTE
    maxfsize       = 460      # max # of bytes in the info part of I-frame
    frmwsize        = 7         # Common window size
    dl_off_delay    = 6
    lap_type        = 4         # NRM Secondary, two way alternate
    frm_optns3      = 0x0018    # SDLC protocol, interpret XIDs
    fhistsz         = 168       # large trace table
dlpip.1
    prim_addr       = 1, 0xFF   # respond to XID on addr FF
    sec_addr        = 1, 0x01
dlpiu.1
    dl_lwr_module_type      = 10            # SNA
    dl_dlpi_provider_lpa    = -1            # to attach
    dl_dlsap                = 1             # bind address (1, 0x01)
    dl_options              = 0xff00        # overkill on cause/diag
    sp_histsz               = 90            # trace table size
    sp_xid_node_id          = 0x04500001
    sp_options              = 0             # no options is the default
    sp_term_options         = 0x20          # NOTIFY, NO UNBIND
    sp_ext_options          = 0x00
    sp_sess_histsz          = 30
```

Figure 10 The sample configuration file **configs.sna.example**

Special SNA Scenarios

sp_ext_options 0x01 (LU 6.2 Server)

If set, the SNA Server will be an LU 6.2 server. The LU 6.2 server acts as an APPC end node. This bit should not be set for any of the other supported LU types.

sp_ext_options 0x04 (Wrap response table)

The response table contains information about outstanding requests that have not been acknowledged by either the target host or the application. If this bit is set, the SNA Server will cause the response table to wrap when full. It will write over the oldest entry in the table if no entries remain empty. If the bit is not set, the SNA Server will terminate the session when the response table fills.

sp_ext_options 0x08 (PU Support)

When this bit is set, it allows applications to send an NMVT request over the PU-SSCP session.

sp_ext_options 0x10 (Short BIND Response)

When set, this option causes the server to reply with only the command byte and not the entire BIND. Some hosts are unable to handle a non-negotiated BIND response that contains the entire BIND.

```
cdi.mod
    n_minors    = 2
    n_upas      = 2
    n_lpas      = 00
dlpi.mod
    n_minors    = 15
    n_upas      = 15
    n_lpas      = 5
npi.mod
    n_minors    = 200
    n_upas      = 200
    n_lpas      = 2
rsys.mod
    bfrsize         = 950       # max RU size + 9 + 100
    r_unix_ticks    = 5
    r_timeout_ms    = 50
cdip.*
    ix25mode        = 01        # DMA
    ix25mode        = 00        # non-DMA
    hertz           = 9830400
    ix25_clks       = 00        # TxC, RxC both inputs
    baudrate        = 38400
    ix25_chip_type  = 13        # HD64570
    ix25_optns      = 0x0a      # free run, ignore rcv aborts
    mdm_msk         = 3
    maxfrmsz        = 0         # default to rsys bfrsize
cdiu.*
    poll_optns      = 0
    pollhsz         = 110       # poll trace table size
    poll_lap_type   = 4         # nrm_sec
    poll_delay      = 50        # milliseconds
dlpip.*
    n2              = 40        # for NRM sec
    fdte            = 1         # DTE
    maxfsize       = 460      # max # of bytes in the info part of I-frame
    frmwsize        = 7         # Common window size
    dl_off_delay    = 6
    lap_type        = 4         # NRM Secondary, two way alternate
    frm_optns3      = 0x0018    # SDLC protocol, interpret XIDs
    fhistsz         = 168       # large trace table
dlpip.1
    prim_addr       = 1, 0xFF   # respond to XID on addr FF
    sec_addr        = 1, 0x01
dlpiu.1
    dl_lwr_module_type      = 10            # SNA
    dl_dlpi_provider_lpa    = -1            # to attach
    dl_dlsap                = 1             # bind address (1, 0x01)
    dl_options              = 0xff00        # overkill on cause/diag
    sp_histsz               = 90            # trace table size
    sp_xid_node_id          = 0x04500001
    sp_options              = 0             # no options is the default
    sp_term_options         = 0x20          # NOTIFY, NO UNBIND
    sp_ext_options          = 0x00
    sp_sess_histsz          = 30
```

Figure 11 The sample configuration file **configs.sna.example**

Synchronous Communications Clocks

ix25_clks

You must specify the *ix25_clks* clocking options parameter for each serial line. Clocks can be supplied by an external source such as a modem or modem eliminator or generated by the serial chip itself. The sample file configures *ix25_clks* in the `cdip.1` section. It will always be configured in a `cdip.`*n* section.

The following table lists the valid values for this parameter along with an explanation.

Table 1 Line Level Clocking Options

Parm. Value	*Tx Clock*	*Rx Clock*	*Explanation*
0	Input	Input	Used with external modem, which generates both transmit and receive clocks. This is usually the clocking mode used for NRZI and FM modes. These modes derive the receive clock from the received data stream. For this clocking option, the transmit clock is provided by the modem.
1	Output	Input	Use for symmetrical back-to-back hookups, where each system generates its own transmit clock and drives the other system's receive clock. The baud rate generator on the chip provides the transmit clock. This option may be used for NRZI or FM modes if the modem is optioned to pay attention to the clock output (usually not the case).
2	Output	From Tx Clk	Used when it is necessary to provide all clocking for the interface and provide the clock on a single output pin. The baud rate generator on the chip provides the transmit clock and the transmit clock is connected to both an output pin and to the receive clock on the chip.
3	Output	Output	Used when it is necessary to provide all clocking for the interface and the serial chip and board have the capability to drive clock outputs on both the transmit and receive clock pins. The baud rate generator on the chip provides both clocks and both clocks are output on separate pins.
4	From Rx Clk	Input	Used for certain modem sharing units that only provide clocking on the receive clock pin. These devices expect the DTE to use this same clock for transmitted data. The receive clock is an input and the transmit clock is connected to the receive clock on the chip. This is the appropriate setting for X.21 cabling when both clocks should be derived from an external source.

Table 1 Line Level Clocking Options

Parm. Value	*Tx Clock*	*Rx Clock*	*Explanation*
5	From Rx Clk	baud rate gen.	Used in conjunction with NRZI or FM when the transmit clock is to be derived from the received data. No clocks are output. This mode is not used very often since it makes the transmitter's clocking dependent upon the existence of received data.

The following illustrations show how Tx Clk and Rx Clk from the serial interface card should be connected to an RS-232C connector.

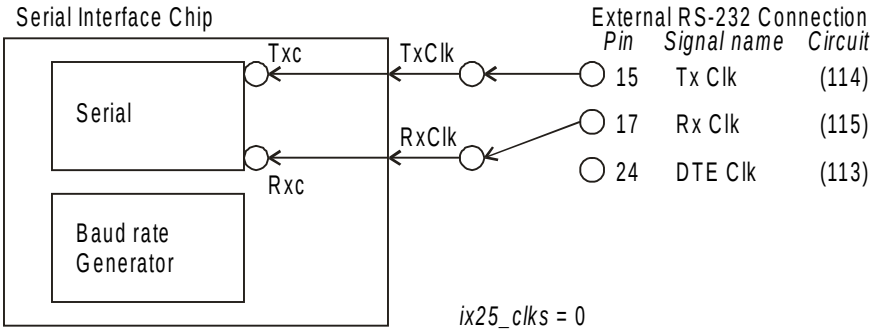


Figure 12 *ix25_clks* = 0

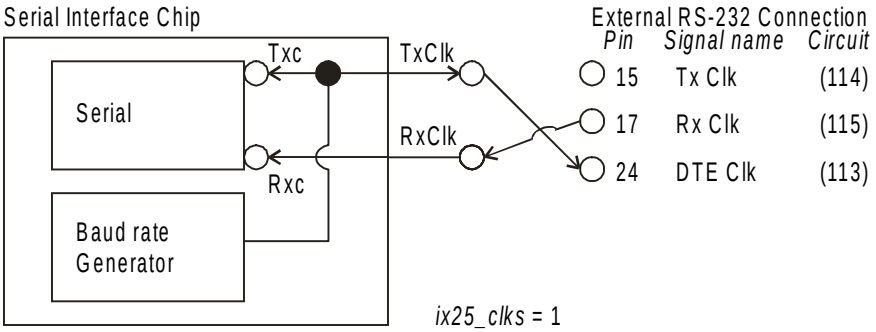


Figure 13 *ix25_clks* = 1

Note 1: The Tx Clk output is usually connected to the DTE Clk pin. Some boards connect this output to the Tx Clk pin.

Note 2: Some boards require external jumpering to allow outputting of TxClk. See the third party board manufacturer's documentation for details.

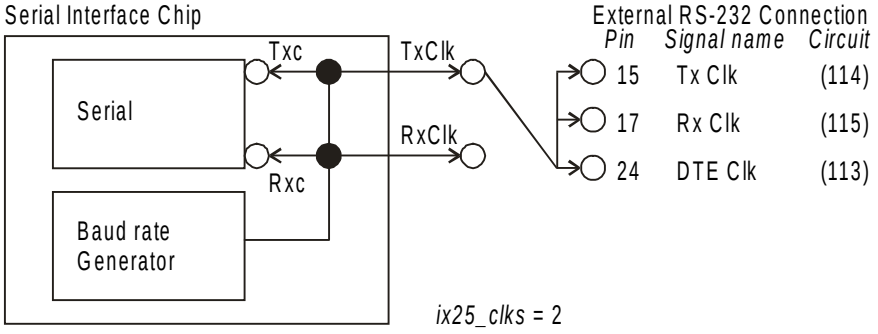


Figure 14 *ix25_clks* = 2

Note 3: Connecting DTE Clk to both Tx Clk and Rx Clk may require using external cabling

See also notes 1 and 2.

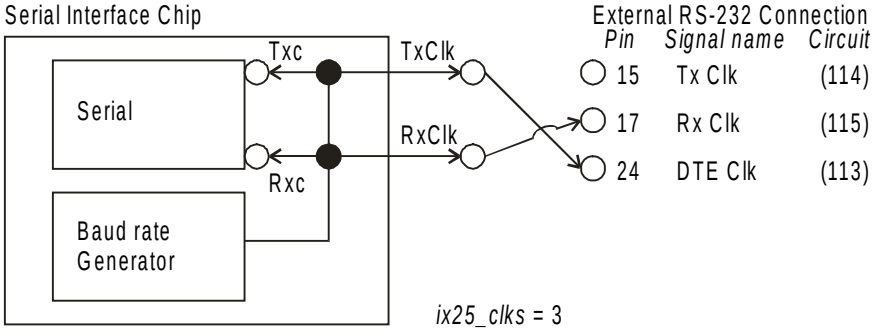


Figure 15 *ix25_clks* = 3

Note 4: It is rare for a board to be capable of producing an RxClk output in this fashion.

See also notes 1 and 2 on the previous page.

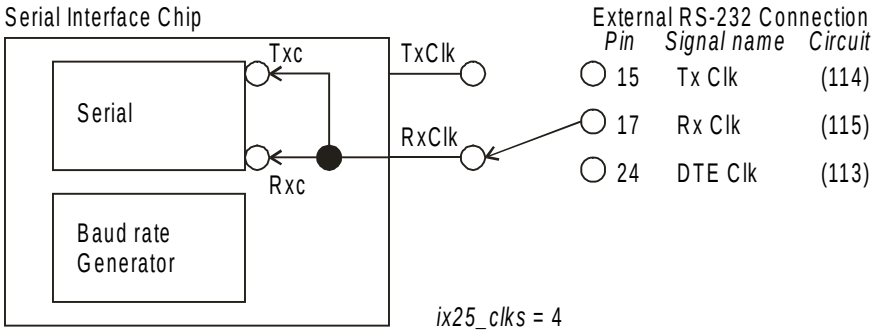


Figure 16 *ix25_clks* = 4

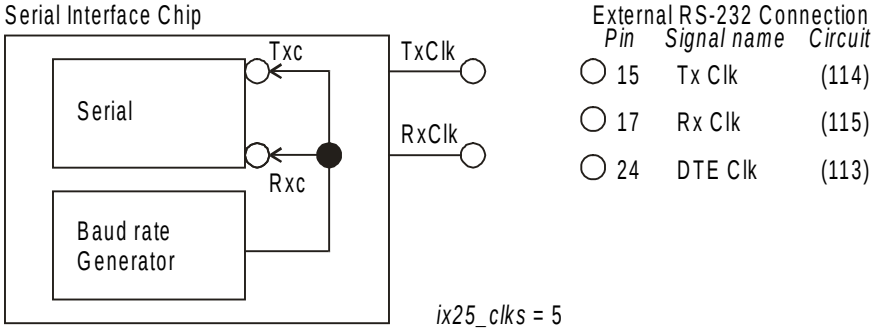


Figure 17 *ix25_clks* = 5

Configuring SDLC/SNA Flow Control

lfc_low

lfc_high

In most situations, flow control between SNA and SDLC won't ever be an issue. However, in high traffic situations, the flow control parameters may indeed have a measurable impact on bottom-line throughput.

By default, *lfc_low* is set to a single window's worth of frames, so that *lfc_low* is equal to *frmwsize*. The default for *lfc_high* is one and one-half windows worth of frames, or *frmwsize* * 1.5. For most applications, this is adequate.

In high-traffic situations, sometimes shifting these numbers slightly higher can increase the throughput on a given line. Trying *lfc_low* at two windows' worth of frames might be a good place to start; set *lfc_high* a few frames higher than *lfc_low* (5 or 6).

The danger in increasing the depth of these queues is that deeper queues can mean longer delays between receipt of a frame in SDLC and delivery of that frame to the appropriate module. This could result in delays when trying to manage SNA pacing, for example. Subsequently, these parameters should be increased with great care, and with attention to the possible negative side-effects.

3

Using NPI/SNA API

The NPI/SNA API Model

The NPA/SNA API provides a layer of “insulation” between the SNA application developer and the details of the SNA protocol. The NPI/SNA API uses a non-blocking approach to protocol operations, accepting requests and calling user-supplied callback routines when events occur.

The core of the API is in the routine *sna_poll_retry()*, which processes protocol events and calls the user supplied routines as appropriate.

Summary of Routines

Initializing/Shutting Down

sna_init_log()	Allows selection of logging options and logfile name.
sna_open()	Opens an SNA data stream and sets up the API resources for managing it.
sna_set_read_bfr()	Sets up the buffer the NPI/SNA API will use for incoming data.
sna_set_buffering_mode()	Configures how much data the API will receive before calling the user's *usr_read_complete()* routine.
sna_close()	Closes down an SNA data stream and frees the API resources.

Connecting

sna_start_connect_lu()	Initiates a connection from a local LU to a remote LU.
usr_connected()	Called when a connection is established.
usr_disconnected()	Called if a connection fails to be established or if an existing connection has been terminated.

Data Transfer

sna_start_write(), sna_sslu_start_write()

Initiates a write operation. The *sna_start_write()* writes to the LU-LU session, while *sna_sslu_start_write()* writes to the SS-LU session.

usr_write_complete(), usr_sslu_write_complete()

Called when the write has completed. The SSLU variant is called when a write to the SS-LU session has completed.

usr_read_complete(), usr_sslu_read_complete()

Called when a buffer has been received. *usr_read_complete()* is called for data received on the LU-LU session, while *usr_sslu_read_complete()* is called for data received on the SS-LU session.

Protocol Features

sna_send_request()	Sends an SNA request to the host.
usr_request_complete()	Called to provide the results of a previously-submitted request.
usr_bid()	Invoked when the SNA host requests to begin a bracket.
usr_host_event()	Invoked for a variety of events caused by host activity.

Utility Routines

sna_get_api_state()	Allows the application to monitor the state of the API.
sna_log_error()	Allows the application to write errors to the log.
usr_log_error_complete()	Called to indicate the success or failure of the error logging.
sna_poll_retry()	Runs the main polling loop. Allows the API to manage I/O and calluser-supplied callbacks.