

# NPI/X.25

## User's Guide

Document Version 1.0

October 2001

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 exist in this document.

Rsystem and GCOM are registered trademarks 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 Gerriets 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

---

<b>SECTION 1</b>	<b>7</b>	<b><i>Configuring NPI/X.25</i></b>
	9	Components of an X.25 Stack
	11	Getting X.25 Configured
	11	<b><i>Layout of the Configs Files</i></b>
	13	<b><i>Unlisted Parameters</i></b>
	13	<b><i>Multiple Occurances of the Same Parameter</i></b>
<b>SECTION 2</b>	<b>15</b>	<b><i>Configuring NPI/X.25: common options</i></b>
	17	Data Sizes
	17	maxpsize
	17	dfltysize
	17	maxfsize
	17	maxfrmsz
	17	bfrsize
	19	Data Window Sizes
	19	frmwsz
	19	dfltwsz
	19	maxwsz
	21	DTE or DCE
	21	<b><i>Frame Level DTE/DCE</i></b>
	21	frm_optns3
	21	fdte
	21	<b><i>Packet Level DTE/DCE</i></b>
	21	pdte
	23	Setting Default Facilities
	23	<b><i>Important Variations</i></b>
	23	<b><i>Critical Parameters</i></b>
	23	mn_ioctl_count
	23	facils
	25	Timers
	25	<b><i>Selecting Timer Values</i></b>
	25	<b><i>Some Important Timers</i></b>
	25	T1_timer
	25	T2_timer
	25	T3_timer
	25	Additional timers
	27	Virtual Circuit Ranges
	27	<b><i>Virtual Circuit Range Parameters</i></b>
	27	lppvc, hipvc
	27	losvc, hisvc

27                    *Configuring a Large Number of Virtual Circuits*  
29                    *Synchronous Communications Clocks*  
29                    *ix25\_clks*

**SECTION 3**

**33**                    *Configuring Multiple Lines*

35                    Configuring Resources

35                    n\_upas

35                    n\_minors

35                    n\_lpas

37                    Routing Outgoing Calls

37                    *Major Differences from Standard Routing*

37                    *Important Parameters*

37                    npi\_route\_id

39                    npi\_route\_addr

39                    mn\_upa

39                    npi\_lcn (PVC's only)

**SECTION 4**

**41**                    *Using the NPI/X.25 API*

42

43                    Overview of the API

45                    A Selection of NPI API Routines

45                    *Initialization and Bind*

45                    *Connecting*

45                    *Listening*

45                    *Data Transfer*

45                    *Disconnect and Close*

# 1

---

## Configuring NPI/X.25

Once the STREAMS Protocol Drivers package is installed, the system must be configured for the communications it will perform. In most cases, configuration is a straightforward process. In more complex scenarios, some extensive modifications to the provided configuration files may be required.

## Components of an X.25 Stack

A GCOM STREAMS X.25 protocol stack has four primary components: the line driver, the STREAMS LAPB module, the STREAMS X.25 module, and the NPI 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 LAPB module, handles frame-level events and maintains a reliable point-to-point connection.

The third component, the STREAMS X.25 module, handles the packet-layer events, managing network routing and end-to-end delivery.

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

An application program will use the NPI 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.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2
npi.mod
    n_minors      = 16
    n_upas       = 16
    n_lpas       = 2
rsys.mod
    bfrsize      = 540          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.1
    ix25mode     = 1          # DMA
    ix25mode     = 0          # non-DMA
    ix25wrapflg = 00         # no loopback
    ix25_chip_type = 13
    hertz        = 9830400
    ix25_clks    = 00         # TxC, RxC both inputs
    baudrate     = 9600
    wd_timer     = 5000       # watchdog timer in milli-sec
    mdm_msk      = 0x03      # DTR & RTS, no rcvd mdm sigs
    maxfrmsz     = 520
cdiu.*
    poll_lap_type = 8          # lapb
dlpip.1
    fdte         = 1          # DTE
    fwrapflg     = 00         # no loopback
    frm_optns2   = 0x62
    frm_optns3   = 0x0020
    fhistsz      = 166       # trace table size
    maxfsize     = 516
    frmwsz       = 7
    T1_timer     = 3
    T2_timer     = 10300
    T3_timer     = 0
dlpiu.1
    pwrapflg     = 00         # no loopback
    lopvc        = 00         # no PVCs
    hipvc        = 00         # no PVCs
    losvc        = 1          # SVC range
    hisvc        = 16         # for production
    dfltwsz     = 2
    maxwsz      = 7
    maxpsz     = 512
    dfltpsz     = 128
    pdte        = 1          # DTE
    std_version  = 1984
npi.mn_ioctl.1
    mn_type      = 0x15       # MN_IOCTL
    mn_upa       = -1        # LPA of pkt lvl
    mn_ioctl_type = 1        # NPI_ADD_ROUTE
    mn_ioctl_count = 72      # size of route config params
    npi_route_id = 100       # unique id
    npi_route_order = 00
    npi_route_addr = i*i     # wildcard address
npi.mn_ioctl.2
    mn_type      = 0x15       # MN_IOCTL
    mn_upa       = -1        # LPA of pkt lvl
    mn_ioctl_type = 7        # NPI_CALL_FACILS
    mn_ioctl_count = 2       # # bytes of facilS
    facilS       = 0x02 0xAA # thrupt class 9600

```

**Figure 1** A sample X.25 configuration file

## Getting X.25 Configured

Setting up an X.25 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/x25` 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 LAPB driver. In **configs.example**, `cdiu.*` handles all `cdiu` subsections, while `dlpip.1` configures a specific LAPB station.

The `dlpiu` and `npi.mn_ioctl` sections configure the STREAMS X.25 driver. In **configs.example**, the `dlpiu.1`, `npi.mn_ioctl.1`, and `npi.mn_ioctl.2` configure packet-level X.25.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2
npi.mod
    n_minors      = 16
    n_upas       = 16
    n_lpas       = 2
rsys.mod
    bfrsize      = 540          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.1
    ix25mode     = 1          # DMA
    ix25mode     = 0          # non-DMA
    ix25wrapflg = 00         # no loopback
    ix25_chip_type = 13
    hertz        = 9830400
    ix25_clks    = 00         # TxC, RxC both inputs
    baudrate     = 9600
    wd_timer     = 5000       # watchdog timer in milli-sec
    mdm_msk     = 0x03       # DTR & RTS, no rcvd mdm sigs
    maxfrmsz    = 520
cdiu.*
    poll_lap_type = 8          # lapb
dlpip.1
    fdte         = 1          # DTE
    fwrapflg    = 00         # no loopback
    frm_optns2   = 0x62
    frm_optns3   = 0x0020
    fhistsz     = 166       # trace table size
    maxfsize    = 516
    frmwsz      = 7
    T1_timer    = 3
    T2_timer    = 10300
    T3_timer    = 0
dlpiu.1
    pwrapflg    = 00         # no loopback
    lopvc       = 00         # no PVCs
    hipvc       = 00         # no PVCs
    losvc       = 1          # SVC range
    hisvc       = 16         # for production
    dfltwsz     = 2
    maxwsz      = 7
    maxpsz     = 512
    dfltpsz     = 128
    pdte        = 1          # DTE
    std_version = 1984
npi.mn_ioctl.1
    mn_type     = 0x15       # MN_IOCTL
    mn_upa      = -1         # LPA of pkt lvl
    mn_ioctl_type = 1       # NPI_ADD_ROUTE
    mn_ioctl_count = 72     # size of route config params
    npi_route_id = 100      # unique id
    npi_route_order = 00
    npi_route_addr = i*i    # wildcard address
npi.mn_ioctl.2
    mn_type     = 0x15       # MN_IOCTL
    mn_upa      = -1         # LPA of pkt lvl
    mn_ioctl_type = 7       # NPI_CALL_FACILS
    mn_ioctl_count = 2      # # bytes of facilS
    facilS     = 0x02 0xAA  # thrupt class 9600

```

**Figure 2** A sample X.25 configuration file

### *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/X.25: common options

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2
npi.mod
    n_minors      = 16
    n_upas       = 16
    n_lpas       = 2
rsys.mod
    bfrsize      = 540          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.1
    ix25mode     = 1          # DMA
    ix25mode     = 0          # non-DMA
    ix25wrapflg = 00         # no loopback
    ix25_chip_type = 13
    hertz        = 9830400
    ix25_clks    = 00         # TxC, RxC both inputs
    baudrate     = 9600
    wd_timer     = 5000       # watchdog timer in milli-sec
    mdm_msk     = 0x03       # DTR & RTS, no rcvd mdm sigs
    maxfrmsz    = 520
cdiu.*
    poll_lap_type = 8          # lapb
dlpip.1
    fdte         = 1          # DTE
    fwrapflg    = 00         # no loopback
    frm_optns2   = 0x62
    frm_optns3   = 0x0020
    fhistsz     = 166       # trace table size
    maxfsize    = 516
    frmwsz      = 7
    T1_timer    = 3
    T2_timer    = 10300
    T3_timer    = 0
dlpiu.1
    pwrapflg    = 00         # no loopback
    lopvc       = 00         # no PVCs
    hipvc       = 00         # no PVCs
    losvc       = 1          # SVC range
    hisvc       = 16         # for production
    dfltwsz     = 2
    maxwsz      = 7
    maxpsz     = 512
    dfltpsz     = 128
    pdte        = 1          # DTE
    std_version = 1984
npi.mn_ioctl.1
    mn_type     = 0x15       # MN_IOCTL
    mn_upa      = -1        # LPA of pkt lvl
    mn_ioctl_type = 1        # NPI_ADD_ROUTE
    mn_ioctl_count = 72     # size of route config params
    npi_route_id = 100      # unique id
    npi_route_order = 00
    npi_route_addr = i*i    # wildcard address
npi.mn_ioctl.2
    mn_type     = 0x15       # MN_IOCTL
    mn_upa      = -1        # LPA of pkt lvl
    mn_ioctl_type = 7        # NPI_CALL_FACILS
    mn_ioctl_count = 2       # # bytes of facils
    facils      = 0x02 0xAA # thrupt class 9600

```

**Figure 3** A sample X.25 configuration file

## Data Sizes

An X.25 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. In most cases, however, the parameters are fairly easily defined.

### **maxsize**

In the `d1piu.n` sections. The maximum packet size for packet level X.25. During the negotiation of options, this is the largest packet size limit that X.25 will allow. It must be a power of two.

### **dfltpsize**

In the `d1piu.n` sections. Default packet size for packet level X.25. This is the packet size limit that X.25 will initially request in negotiations. It must be a power of two.

### **maxfsz**

In the `d1pip.n` sections. Maximum frame size for the link layer (LAPB). Should include space for a packet header. *maxsize*+ 4 should prove adequate.

### **maxfrmsz**

In the `cdip.n` sections. Maximum frame size for the line driver. Includes space for a packet header and frame header. *maxfsz*+ 4 should prove adequate.

### **bfrsize**

In `rsys.mod`. Overall buffer size. Must be at least as large as *maxfrmsz*. We use *maxfrmsz*+ 20 in the example, which allows adequate working room in the buffer for miscellaneous protocol components.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2
npi.mod
    n_minors      = 16
    n_upas       = 16
    n_lpas       = 2
rsys.mod
    bfrsize      = 540          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.1
    ix25mode     = 1          # DMA
    ix25mode     = 0          # non-DMA
    ix25wrapflg = 00         # no loopback
    ix25_chip_type = 13
    hertz        = 9830400
    ix25_clks    = 00         # TxC, RxC both inputs
    baudrate     = 9600
    wd_timer     = 5000       # watchdog timer in milli-sec
    mdm_msk      = 0x03      # DTR & RTS, no rcvd mdm sigs
    maxfrmsz     = 520
cdiu.*
    poll_lap_type = 8          # lapb
dlpip.1
    fdte         = 1          # DTE
    fwrapflg    = 00         # no loopback
    frm_optns2   = 0x62
    frm_optns3   = 0x0020
    fhistsz     = 166        # trace table size
    maxfsz      = 516
    frmwsz      = 7
    T1_timer    = 3
    T2_timer    = 10300
    T3_timer    = 0
dlpiu.1
    pwrapflg    = 00         # no loopback
    lopvc       = 00         # no PVCs
    hipvc       = 00         # no PVCs
    losvc       = 1          # SVC range
    hisvc       = 16         # for production
    dfltwsz     = 2
    maxwsz      = 7
    maxpsz     = 512
    dfltpsz     = 128
    pdte        = 1          # DTE
    std_version = 1984
npi.mn_ioctl.1
    mn_type     = 0x15        # MN_IOCTL
    mn_upa      = -1         # LPA of pkt lvl
    mn_ioctl_type = 1        # NPI_ADD_ROUTE
    mn_ioctl_count = 72      # size of route config params
    npi_route_id = 100       # unique id
    npi_route_order = 00
    npi_route_addr = i*i     # wildcard address
npi.mn_ioctl.2
    mn_type     = 0x15        # MN_IOCTL
    mn_upa      = -1         # LPA of pkt lvl
    mn_ioctl_type = 7        # NPI_CALL_FACILS
    mn_ioctl_count = 2       # # bytes of facils
    facils      = 0x02 0xAA  # thrupt class 9600

```

**Figure 4** A sample X.25 configuration file

## 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.

### **frmwsiz**

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

### **dfltwsize**

Default number of packets for the packet-level (X.25) window. The default number will be the number sent as an initial requested window size during negotiation of connection options. Configured in the `dlpiu.n` sections.

### **maxwsiz**

Largest window size limit which can be supported for the packet-level (X.25) window. X.25 will not allow a call to negotiate a window size that exceeds this number of packets. Configured in the `dlpiu.n` sections.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2
npi.mod
    n_minors      = 16
    n_upas       = 16
    n_lpas       = 2
rsys.mod
    bfrsize      = 540          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.1
    ix25mode     = 1          # DMA
    ix25mode     = 0          # non-DMA
    ix25wrapflg = 00         # no loopback
    ix25_chip_type = 13
    hertz        = 9830400
    ix25_clks    = 00         # TxC, RxC both inputs
    baudrate     = 9600
    wd_timer     = 5000       # watchdog timer in milli-sec
    mdm_msk      = 0x03      # DTR & RTS, no rcvd mdm sigs
    maxfrmsz     = 520
cdiu.*
    poll_lap_type = 8          # lapb
dlpip.1
    fdte         = 1          # DTE
    fwrapflg     = 00         # no loopback
    frm_optns2   = 0x62
    frm_optns3   = 0x0020
    fhistsz      = 166       # trace table size
    maxfsize     = 516
    frmwsz       = 7
    T1_timer     = 3
    T2_timer     = 10300
    T3_timer     = 0
dlpiu.1
    pwrapflg     = 00         # no loopback
    lopvc        = 00         # no PVCs
    hipvc        = 00         # no PVCs
    losvc        = 1          # SVC range
    hisvc        = 16         # for production
    dfltwsz     = 2
    maxwsz       = 7
    maxpsz      = 512
    dfltpsz     = 128
    pdte        = 1          # DTE
    std_version  = 1984
npi.mn_ioctl.1
    mn_type      = 0x15       # MN_IOCTL
    mn_upa       = -1        # LPA of pkt lvl
    mn_ioctl_type = 1         # NPI_ADD_ROUTE
    mn_ioctl_count = 72      # size of route config params
    npi_route_id = 100       # unique id
    npi_route_order = 00
    npi_route_addr = i*i     # wildcard address
npi.mn_ioctl.2
    mn_type      = 0x15       # MN_IOCTL
    mn_upa       = -1        # LPA of pkt lvl
    mn_ioctl_type = 7         # NPI_CALL_FACILS
    mn_ioctl_count = 2       # # bytes of facilS
    facilS       = 0x02 0xAA # thrupt class 9600

```

**Figure 5** A sample X.25 configuration file

## DTE or DCE

X.25 supports the concept of DTE and DCE on two levels: at the frame level, for the point-to-point connection, and at the packet level, for the network connection. The *fdte* and *pdte* parameters control this setting, and a bit in *frm\_optns3* allows for automatic handling of frame-level DTE or DCE.

### *Frame Level DTE/DCE*

DTE or DCE can be manually set using the *fdte* parameter, if a user requires this functionality. However, at the frame level, LAPB can detect what role the remote system is playing and automatically configure itself properly. This is controlled by a bit in the *frm\_optns3* parameter.

#### **frm\_optns3**

The 0x020 bit in *frm\_optns3* turns on a special DTE/DCE handling mechanism that detects what the remote host has configured itself to be and responds with a complimentary configuration. This automatic frame-level DTE/DCE mechanism is highly recommended.

#### **fdte**

In the `d1pip.n` sections. Frame level DTE or DCE. Set to 1 for DTE or 0 for DCE.

### *Packet Level DTE/DCE*

Packet-level DTE or DCE may also be set manually using the *pdte* parameter. If the *pdte* parameter is not present, the setting for packet-level DTE/DCE will be derived from the frame-level DTE/DCE.

#### **pdte**

In the `d1piu.n` sections. Packet level DTE or DCE. Set to 1 for DTE or 0 for DCE.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2
npi.mod
    n_minors      = 16
    n_upas       = 16
    n_lpas       = 2
rsys.mod
    bfrsize      = 540          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.1
    ix25mode     = 1          # DMA
    ix25mode     = 0          # non-DMA
    ix25wrapflg = 00         # no loopback
    ix25_chip_type = 13
    hertz        = 9830400
    ix25_clks    = 00         # TxC, RxC both inputs
    baudrate     = 9600
    wd_timer     = 5000       # watchdog timer in milli-sec
    mdm_msk      = 0x03      # DTR & RTS, no rcvd mdm sigs
    maxfrmsz    = 520
cdiu.*
    poll_lap_type = 8          # lapb
dlpip.1
    fdte         = 1          # DTE
    fwrapflg    = 00         # no loopback
    frm_optns2   = 0x62
    frm_optns3   = 0x0020
    fhistsz     = 166        # trace table size
    maxfsz      = 516
    frmwsz      = 7
    T1_timer    = 3
    T2_timer    = 10300
    T3_timer    = 0
dlpiu.1
    pwrapflg    = 00         # no loopback
    lopvc       = 00         # no PVCs
    hipvc       = 00         # no PVCs
    losvc       = 1          # SVC range
    hisvc       = 16         # for production
    dfltwsz     = 2
    maxwsz      = 7
    maxpsz     = 512
    dfltpsz     = 128
    pdte        = 1          # DTE
    std_version = 1984
npi.mn_ioctl.1
    mn_type     = 0x15        # MN_IOCTL
    mn_upa      = -1         # LPA of pkt lvl
    mn_ioctl_type = 1        # NPI_ADD_ROUTE
    mn_ioctl_count = 72      # size of route config params
    npi_route_id = 100       # unique id
    npi_route_order = 00
    npi_route_addr = i*i     # wildcard address
npi.mn_ioctl.2
    mn_type     = 0x15        # MN_IOCTL
    mn_upa      = -1         # LPA of pkt lvl
    mn_ioctl_type = 7        # NPI_CALL_FACILS
    mn_ioctl_count = 2       # # bytes of facilS
    facilS      = 0x02 0xAA  # thrupt class 9600

```

**Figure 6** A sample X.25 configuration file

## Setting Default Facilities

Management I/O controls are used to set default facilities for outgoing calls. Setting up a management I/O control requires filling in the parameters on an appropriate `npi.mn_ioctl` template.

The sample file at left uses two different `npi.mn_ioctl` templates. The section `npi.mn_ioctl.2` sets up two bytes of default facilities using the proper template.

### *Important Variations*

*additional facilities* A configuration that sets different or additional facilities would replace the *mn\_ioctl\_count* and *facils* parameters from this sample to reflect the new facilities.

*section naming* If a *configs* file has no `npi.mn_ioctl.1` section, *Gcom\_monitor* will not process any other `npi.mn_ioctl.n` sections. *Gcom\_monitor* processes the `npi.mn_ioctl.n` sections one at a time, in the sequence indicated by *n*. If the sequence is broken, *Gcom\_monitor* will stop at the break.

### *Critical Parameters*

#### **mn\_ioctl\_count**

*mn\_ioctl\_count* is the number of bytes of facilities included in the request. If additional facilities are specified, the number will be larger. The sample at right specifies 2 bytes of facilities.

#### **facils**

The *facils* parameter is the actual series of facilities specified. The sample at right specifies two bytes of facilities, requesting a throughput class specification of 9600 baud both sending and receiving.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2
npi.mod
    n_minors      = 16
    n_upas       = 16
    n_lpas       = 2
rsys.mod
    bfrsize      = 540          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.1
    ix25mode     = 1          # DMA
    ix25mode     = 0          # non-DMA
    ix25wrapflg  = 00        # no loopback
    ix25_chip_type = 13
    hertz        = 9830400
    ix25_clks    = 00        # TxC, RxC both inputs
    baudrate     = 9600
    wd_timer     = 5000      # watchdog timer in milli-sec
    mdm_msk      = 0x03     # DTR & RTS, no rcvd mdm sigs
    maxfrmsz    = 520
cdiu.*
    poll_lap_type = 8        # lapb
dlpip.1
    fdte         = 1        # DTE
    fwrapflg    = 00       # no loopback
    frm_optns2   = 0x62
    frm_optns3   = 0x0020
    fhistsz     = 166     # trace table size
    maxfsz      = 516
    frmwsz      = 7
    T1_timer    = 3
    T2_timer    = 10300
    T3_timer    = 0
dlpiu.1
    pwrapflg    = 00       # no loopback
    lopvc       = 00       # no PVCs
    hipvc       = 00       # no PVCs
    losvc       = 1        # SVC range
    hisvc       = 16       # for production
    dfltwsz     = 2
    maxwsz      = 7
    maxpsz     = 512
    dfltpsz     = 128
    pdte        = 1        # DTE
    std_version = 1984
npi.mn_ioctl.1
    mn_type     = 0x15     # MN_IOCTL
    mn_upa      = -1      # LPA of pkt lvl
    mn_ioctl_type = 1     # NPI_ADD_ROUTE
    mn_ioctl_count = 72   # size of route config params
    npi_route_id = 100    # unique id
    npi_route_order = 00
    npi_route_addr = i*i   # wildcard address
npi.mn_ioctl.2
    mn_type     = 0x15     # MN_IOCTL
    mn_upa      = -1      # LPA of pkt lvl
    mn_ioctl_type = 7     # NPI_CALL_FACILS
    mn_ioctl_count = 2    # # bytes of facilS
    facilS      = 0x02 0xAA # thrupt class 9600

```

**Figure 7** A sample X.25 configuration file

# Timers

Synchronous protocols rely on various timers to properly coordinate communications. The default values of these timers will be well within tolerances for most environments.

## *Selecting Timer Values*

GCOM's timer settings allow for two degrees of precision. For many installations, timers can be set to the nearest second. Values from 0 to 10,000 are interpreted in this way, as the number of whole seconds. For values in excess of 10,000, subtract 10,000 to get the number of milliseconds. In other words, the value 13,500 would be interpreted as  $13,500 - 10,000 = 3,500$  milliseconds, or 3.5 seconds.

## *Some Important Timers*

### **T1\_timer**

The *T1\_timer* specifies how long LAPB should wait before retransmitting frames. Set in the `dlpip.n` sections.

### **T2\_timer**

The *T2\_timer* specifies how long LAPB should wait for outbound traffic before sending an RR to acknowledge received data. Set in the `dlpip.n` sections.

### **T3\_timer**

The *T3\_timer* specifies how long a link can be idle before it is considered "down". Set in the `dlpip.n` sections.

## **Additional timers**

Other timers can be configured in the configuration file. The parameters controlling these timers are described in the *STREAMS Administrator's Guide*. These parameters may employ different conventions when constructing timer values. Consult the *STREAMS Administrator's Guide* for details.

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 2
npi.mod
    n_minors      = 16
    n_upas        = 16
    n_lpas        = 2
rsys.mod
    bfrsize       = 540          # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.1
    ix25mode      = 1           # DMA
    ix25mode      = 0           # non-DMA
    ix25wrapflg   = 00         # no loopback
    ix25_chip_type = 13
    hertz         = 9830400
    ix25_clks     = 00         # TxC, RxC both inputs
    baudrate      = 9600
    wd_timer      = 5000       # watchdog timer in milli-sec
    mdm_msk       = 0x03       # DTR & RTS, no rcvd mdm sigs
    maxfrmsz      = 520
cdiu.*
    poll_lap_type = 8           # lapb
dlpip.1
    fdte          = 1           # DTE
    fwrapflg     = 00         # no loopback
    frm_optns2    = 0x62
    frm_optns3    = 0x0020
    fhistsz      = 166        # trace table size
    maxfsize     = 516
    frmwsz       = 7
    T1_timer     = 3
    T2_timer     = 10300
    T3_timer     = 0
dlpiu.1
    pwrapflg     = 00         # no loopback
    lopvc        = 00         # no PVCs
    hipvc        = 00         # no PVCs
    losvc        = 1          # SVC range
    hisvc        = 16         # for production
    dfltwsz     = 2
    maxwsz      = 7
    maxpsz     = 512
    dfltpsz     = 128
    pdte        = 1           # DTE
    std_version  = 1984
npi.mn_ioctl.1
    mn_type      = 0x15        # MN_IOCTL
    mn_upa       = -1         # LPA of pkt lvl
    mn_ioctl_type = 1         # NPI_ADD_ROUTE
    mn_ioctl_count = 72       # size of route config params
    npi_route_id = 100        # unique id
    npi_route_order = 00
    npi_route_addr = i*i      # wildcard address
npi.mn_ioctl.2
    mn_type      = 0x15        # MN_IOCTL
    mn_upa       = -1         # LPA of pkt lvl
    mn_ioctl_type = 7         # NPI_CALL_FACILS
    mn_ioctl_count = 2        # # bytes of facils
    facils       = 0x02 0xAA  # thrupt class 9600

```

**Figure 8** A sample X.25 configuration file

## Virtual Circuit Ranges

### *Virtual Circuit Range Parameters*

The virtual circuit range parameters set the channel ranges for Permanent Virtual Circuits (PVCs) and Switched Virtual Circuits (SVC). The sample file at left configures these ranges in the `dlpiu.1` section, with the parameters *lopvc*, *hipvc*, *losvc*, and *hisvc*.

#### **lppvc, hipvc**

The lowest and highest PVC numbers to use. Configured in the `dlpiu.n` sections.

#### **losvc, hisvc**

The lowest and highest SVC number to use. Configured in the `dlpiu.n` sections.

### *Configuring a Large Number of Virtual Circuits*

Configuring a large number of virtual circuits requires an adequate pool of STREAMS resources to support the circuits. STREAMS needs a minor device (and subsequently a UPA) at the NPI level for each virtual circuit. This means that `npi.mod`'s *n\_minors* and *n\_upas* should both be set to a value equal to the total number of virtual circuits desired.

The sample file at left is set up to support 16 virtual circuits. If the *hisvc* parameter were increased to 200, then `npi.mod`'s *n\_minors* and *n\_upas* would also need to be set to 200.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2
npi.mod
    n_minors      = 16
    n_upas       = 16
    n_lpas       = 2
rsys.mod
    bfrsize      = 540          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.1
    ix25mode     = 1          # DMA
    ix25mode     = 0          # non-DMA
    ix25wrapflg  = 00        # no loopback
    ix25_chip_type = 13
    hertz        = 9830400
    ix25_clks    = 00        # TxC, RxC both inputs
    baudrate     = 9600
    wd_timer     = 5000      # watchdog timer in milli-sec
    mdm_msk      = 0x03     # DTR & RTS, no rcvd mdm sigs
    maxfrmsz    = 520
cdiu.*
    poll_lap_type = 8        # lapb
dlpip.1
    fdte         = 1        # DTE
    fwrapflg    = 00       # no loopback
    frm_optns2   = 0x62
    frm_optns3   = 0x0020
    fhistsz     = 166     # trace table size
    maxfsize    = 516
    frmwsz      = 7
    T1_timer    = 3
    T2_timer    = 10300
    T3_timer    = 0
dlpiu.1
    pwrapflg    = 00       # no loopback
    lopvc       = 00       # no PVCs
    hipvc       = 00       # no PVCs
    losvc       = 1        # SVC range
    hisvc       = 16       # for production
    dfltwsz     = 2
    maxwsz      = 7
    maxpsz     = 512
    dfltpsz     = 128
    pdte        = 1        # DTE
    std_version = 1984
npi.mn_ioctl.1
    mn_type     = 0x15     # MN_IOCTL
    mn_upa      = -1      # LPA of pkt lvl
    mn_ioctl_type = 1     # NPI_ADD_ROUTE
    mn_ioctl_count = 72   # size of route config params
    npi_route_id = 100    # unique id
    npi_route_order = 00
    npi_route_addr = i*i  # wildcard address
npi.mn_ioctl.2
    mn_type     = 0x15     # MN_IOCTL
    mn_upa      = -1      # LPA of pkt lvl
    mn_ioctl_type = 7     # NPI_CALL_FACILS
    mn_ioctl_count = 2    # # bytes of facilS
    facilS      = 0x02 0xAA # thrupt class 9600

```

**Figure 9** A sample X.25 configuration file

## 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..

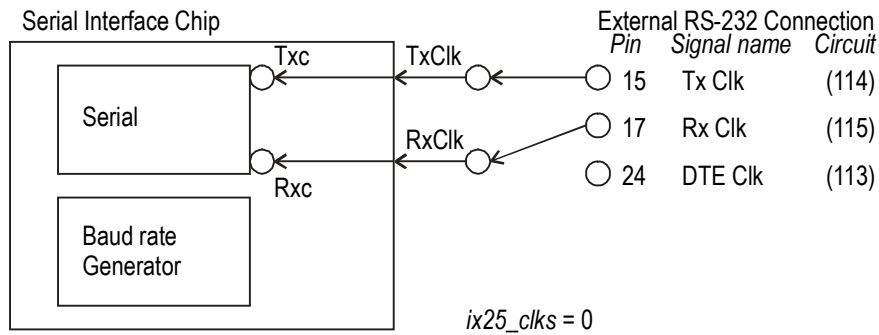
**Table 1** Line Level Clocking Options

<i>Parm. Value</i>	<i>Tx Clock</i>	<i>Rx Clock</i>	<i>Explanation</i>
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.

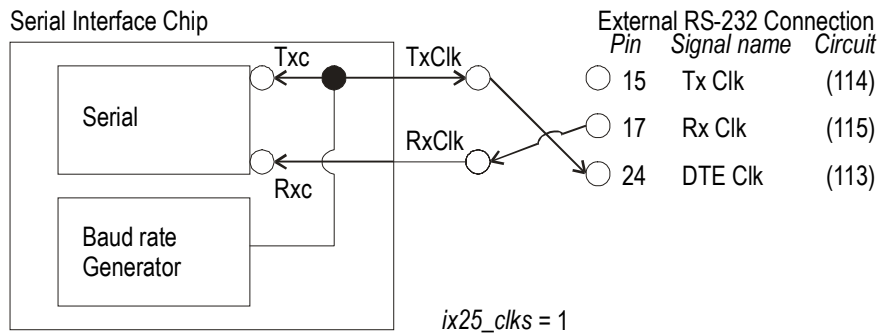
**Table 1** Line Level Clocking Options

<i>Parm. Value</i>	<i>Tx Clock</i>	<i>Rx Clock</i>	<i>Explanation</i>
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.



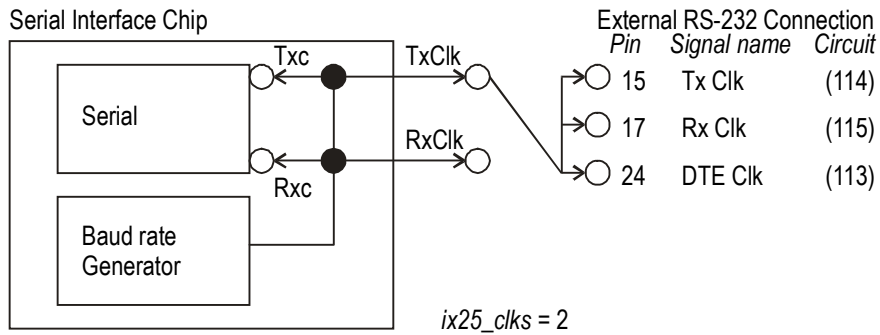
**Figure 10**  $ix25\_clks = 0$



**Figure 11**  $ix25\_clks = 1$

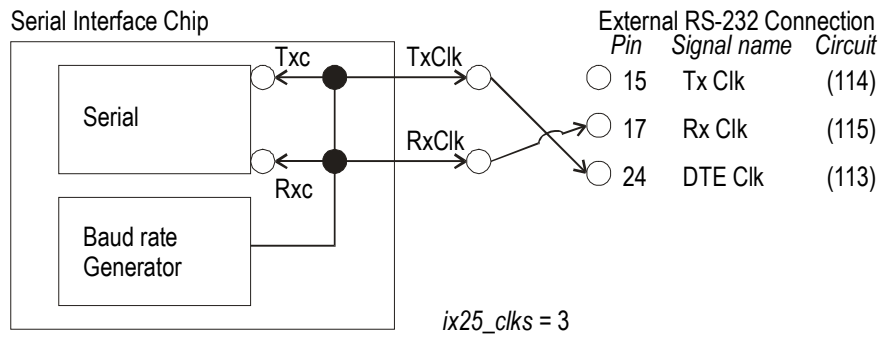
**Note 1:** The Tx Clk output is usually connected to the DTE Clk pin. Some boards connect this output to the Tx Clk pin.

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



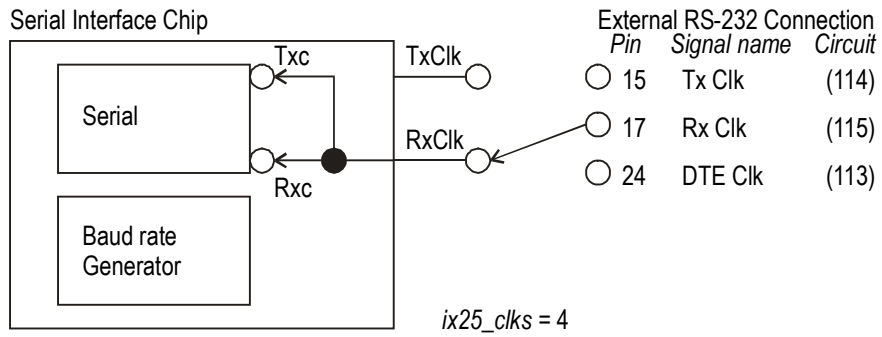
**Figure 12**  $ix25\_clks = 2$

**Note 3:** Connecting DTE Clk to both Tx Clk and Rx Clk may require using external cabling. See also notes 1 and 2.

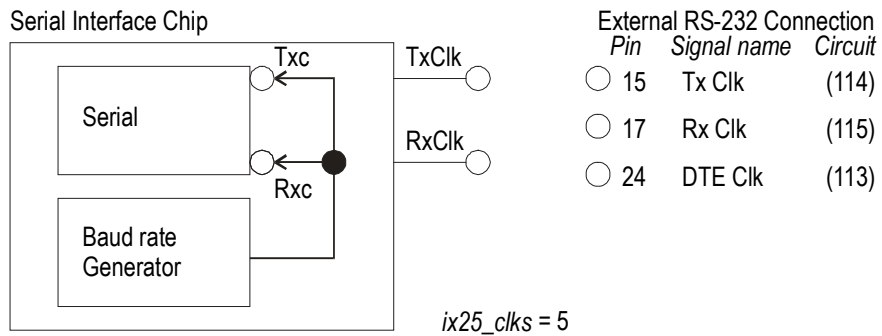


**Figure 13**  $ix25\_clks = 3$

**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 14**  $ix25\_clks = 4$



**Figure 15**  $ix25\_clks = 5$

# 3

---

## Configuring Multiple Lines

Setting up multiple lines takes two steps. The first step configures the resources necessary for the STREAMS facility to support multiple lines. The second step configures routing of outbound calls across these lines.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00

dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2

npi.mod
    n_minors      = 200
    n_upas       = 200
    n_lpas       = 2

rsys.mod
    bfrsize      = 460          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50

.
.
.

npi.mn_ioctl.2
    mn_type      = 0x15          # MN_IOCTL
    mn_upa       = -2           # LPA of pkt lvl
    mn_ioctl_type = 1           # NPI_ADD_ROUTE
    mn_ioctl_count = 72         # size of route config params
    npi_route_id = 101         # unique id
    npi_route_order = 00
    npi_route_addr = i1314*i   # wildcard address
    npi_cost_per_pkt_lvl = 00
    npi_cost_per_call_2 = 00
    npi_cost_per_call = 00
    npi_priority = 00
    npi_rsvd1    = 00
    npi_rsvd2    = 00
    npi_rsvd3    = 00

npi.mn_ioctl.1
    mn_type      = 0x15          # MN_IOCTL
    mn_upa       = -1           # LPA of pkt lvl
    mn_ioctl_type = 1           # NPI_ADD_ROUTE
    mn_ioctl_count = 72         # size of route config params
    npi_route_id = 100         # unique id
    npi_route_order = 00
    npi_route_addr = i1312*i   # wildcard address
    npi_cost_per_pkt_lvl = 00
    npi_cost_per_call_2 = 00
    npi_cost_per_call = 00
    npi_priority = 00
    npi_rsvd1    = 00
    npi_rsvd2    = 00
    npi_rsvd3    = 00

```

**Figure 16** A sample X.25 configuration file

## Configuring Resources

The first three sections of typical configs files are the ***cdi.mod***, ***dlpi.mod***, and ***npi.mod*** sections. Each of these sections contains the same set of three parameters: ***n\_minors***, ***n\_upas***, ***n\_lpas***. The values provided in these sections indicate how much of the system's resources ***Gcom\_monitor*** should reserve to support these various aspects of the STREAMS system.

### ***n\_upas***

This is the number of UPAs the module should support. A module's UPA count is the sum of all the upstream connections it will need to support. LAPB, for example, uses DLPI services. If LAPB expects to support two separate packet-level X.25 devices, this count would need to be at least 2.

If those two X.25 modules then expect to support 10 simultaneous connections apiece, the NPI module's ***n\_upas*** would need to be at least 20.

### ***n\_minors***

This is the number of minor devices to create for the specified module. This parameter should be equal to the number of UPAs specified with ***n\_upas***

### ***n\_lpas***

This is the number of LPAs the module should support. A module's LPA count is ordinarily equal to the UPA count of the layer immediately below it. For CDI (and thus ***cdi.mod***), this parameter is always 0 and only included in configuration files so that the value is explicitly set.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00

dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2

npi.mod
    n_minors      = 200
    n_upas       = 200
    n_lpas       = 2

rsys.mod
    bfrsize      = 460          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50

.
.
.

npi.mn_ioctl.2
    mn_type      = 0x15          # MN_IOCTL
    mn_upa       = -2          # LPA of pkt lvl
    mn_ioctl_type = 1          # NPI_ADD_ROUTE
    mn_ioctl_count = 72        # size of route config params
    npi_route_id = 101        # unique id
    npi_route_order = 00
    npi_route_addr = i1314*i   # wildcard address
    npi_cost_per_pkt_lvl = 00
    npi_cost_per_call_2 = 00
    npi_cost_per_call = 00
    npi_priority = 00
    npi_rsvd1    = 00
    npi_rsvd2    = 00
    npi_rsvd3    = 00

npi.mn_ioctl.1
    mn_type      = 0x15          # MN_IOCTL
    mn_upa       = -1          # LPA of pkt lvl
    mn_ioctl_type = 1          # NPI_ADD_ROUTE
    mn_ioctl_count = 72        # size of route config params
    npi_route_id = 100        # unique id
    npi_route_order = 00
    npi_route_addr = i1312*i   # wildcard address
    npi_cost_per_pkt_lvl = 00
    npi_cost_per_call_2 = 00
    npi_cost_per_call = 00
    npi_priority = 00
    npi_rsvd1    = 00
    npi_rsvd2    = 00
    npi_rsvd3    = 00

```

**Figure 17** Excerpt from a sample X.25 configuration file

## Routing Outgoing Calls

Setting up additional lines also requires configuration of call routing. Call routing directs outbound calls across the appropriate line. Routing for X.25 is handled through the NPI's internal routing tables. To set up these routing tables, parameters must be in the appropriate management I/O control template. The example configuration at left sets up routes for two lines. The `npi.mn_ioctl.1` section configures a route to addresses beginning with "1312" and the `npi.mn_ioctl.2` section configures a route to addresses beginning with "1314". Each uses the same basic set of parameters with a few key differences.

### *Major Differences from Standard Routing*

- additional lines* Configuring additional lines requires additional `npi.mn_ioctl.n` sections in the **configs** file. Each is formatted according to the same basic template. The *nm\_upa*, *npi\_route\_id* and *npi\_route\_addr* parameters will change from section to section, and for permanent virtual circuits (PVC's), the *npi\_lcn* will also be unique to a given route.
- section naming* If a **configs** file has no `npi.mn_ioctl.1` section, *Gcom\_monitor* will not process any other `npi.mn_ioctl.n` sections. *Gcom\_monitor* processes the `npi.mn_ioctl.n` sections one at a time, in sequence of their extension *n*. If this sequence is broken, *Gcom\_monitor* will stop at the break; in other words, `npi.mn_ioctl.3` will not be processed unless `npi.mn_ioctl.2` is present.

### *Important Parameters*

#### **npi\_route\_id**

The *npi\_route\_id* is simply a unique number used to identify this routing table entry. The value isn't important beyond serving as a unique index into the routing table.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00

dlpi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 2

npi.mod
    n_minors      = 200
    n_upas       = 200
    n_lpas       = 2

rsys.mod
    bfrsize      = 460          # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50

.
.
.

npi.mn_ioctl.2
    mn_type      = 0x15          # MN_IOCTL
    mn_upa       = -2           # LPA of pkt lvl
    mn_ioctl_type = 1           # NPI_ADD_ROUTE
    mn_ioctl_count = 72         # size of route config params
    npi_route_id = 101         # unique id
    npi_route_order = 00
    npi_route_addr = i1314*i   # wildcard address
    npi_cost_per_pkt_lvl = 00
    npi_cost_per_call_2 = 00
    npi_cost_per_call = 00
    npi_priority = 00
    npi_rsvd1    = 00
    npi_rsvd2    = 00
    npi_rsvd3    = 00

npi.mn_ioctl.1
    mn_type      = 0x15          # MN_IOCTL
    mn_upa       = -1           # LPA of pkt lvl
    mn_ioctl_type = 1           # NPI_ADD_ROUTE
    mn_ioctl_count = 72         # size of route config params
    npi_route_id = 100         # unique id
    npi_route_order = 00
    npi_route_addr = i1312*i   # wildcard address
    npi_cost_per_pkt_lvl = 00
    npi_cost_per_call_2 = 00
    npi_cost_per_call = 00
    npi_priority = 00
    npi_rsvd1    = 00
    npi_rsvd2    = 00
    npi_rsvd3    = 00

```

**Figure 18** Excerpt from a sample X.25 configuration file

## **npi\_route\_addr**

The *npi\_route\_addr* parameter is the most significant part of the routing table entry. *npi\_route\_addr* is a template address used to map outbound calls to a particular line.

More complicated configurations may require specifying part or all of the address for a particular line. When constructing an address pattern, any number will match itself, a ? will match any single character, and a \* will match any sequence of characters.

## **mn\_upa**

*mn\_upa* assigns a line number to the route. This line will be used for calls to addresses matching this routing entry.

Specify the line as the negative of the appropriate LAPB's extension. In the sample file, our `d1p1p.1` LAPB would be specified with -1, while the second line, `d1p1p.2`, would be specified as -2.

Advanced configurations may require more sophisticated techniques for identifying the proper line. Details on alternate methods can be found in the *STREAMS Administrator's Guide*.

## **npi\_lcn (PVC's only)**

The *npi\_lcn* is the logical channel number of a permanent virtual circuit. The principal difference between a permanent virtual circuit (PVC) and a switched virtual circuit (SVC) is that the PVC will have a permanent logical channel number (and a permanent and continuous connection) whereas the SVC will negotiate the logical channel number each time a call is placed.

# 4

---

## Using the NPI/X.25 API

This chapter provides a quick overview of the way X.25 programs employ the facilities of the NPI API and the facilities of the API itself. The first part of the section details the techniques an application should employ to make use of the NPI API. The second section discusses the specific routines provided by the API.

Developers should refer to the NPI API Guide for additional information on the use of the API or the functions it presents.



## Overview of the API

X.25 uses the NPI API to provide access to X.25 communications. Normal sending and receiving of regular data are accomplished just as for any other NPI protocol.

Applications begin by initializing the API's data space and opening a logfile via the API. They open a data stream, then bind that stream to an address. Applications will then either connect to the remote host using a standard NSAP address, or listen on the bound address for incoming calls.

## A Selection of NPI API Routines

### *Initialization and Bind*

- np\_i\_init()* initializes the API's data structures and opens a logfile
- np\_i\_open\_data()* opens a data stream
- np\_i\_bind\_ascii\_nsap()* binds the data stream to an address pattern

### *Connecting*

- np\_i\_connect()* opens a data stream, binds it, and initiates a connection on it
- np\_i\_connect\_req()* initiates a connection on a previously opened and bound data stream
- np\_i\_send\_connect\_req()* initiates a connection but does not wait for the connection to be established before returning

### *Listening*

- np\_i\_listen()* opens a data stream, binds it, and listens for incoming calls
- np\_i\_ext\_listen()* like *np\_i\_listen()*, but allows for facilities

### *Data Transfer*

- np\_i\_read\_data()* simple read routine for managing data transfer
- np\_i\_rcv()* a more complex read routine with provision for data flags
- np\_i\_write\_data()* simple write routine designed to write normal data
- np\_i\_put\_data\_proto()* more complex write routine designed to allow use of flags with the data

### *Disconnect and Close*

- np\_i\_discon\_req()* gracefully shuts down an X.25 connection
- close()* closes the stream and releases its resources