

STREAMS BISYNC

User's Guide

November 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 exist 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 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

SECTION 1	7	<i>Configuring STREAMS Bisync</i>
	9	Components of the Bisync Stack
	11	The Configuration Process
	11	<i>Layout of the File</i>
	11	<i>Unlisted Parameters</i>
	11	<i>Multiple Occurances of the Same Parameter</i>
SECTION 2	13	<i>Configuring Line-level Bisync</i>
	15	Configuring the Line Driver
	17	<i>baudrate—Baud Rate</i>
	19	<i>maxfrmsz—Frame Size</i>
	19	<i>ix25mode</i>
	21	<i>hdx_options</i>
	21	<i>ix25_ll_options</i>
	23	<i>seq_end_dly</i>
	23	<i>lt_dly</i>
	23	<i>ix25wrapflg</i>
	25	<i>ix25_clks—Clock Routing Options for NRZ</i>
SECTION 3	29	<i>Configuring 2780/3780 Bisync</i>
	31	Basic Configuration
	31	<i>Selecting Protocol Type</i>
	33	<i>Configuring Transparent Mode and ASCII Transfer</i>
	33	<i>Emulating a 2780/3780 Primary</i>
	35	<i>Setting Up Routing</i>
	35	<i>Important Parameters: npi_route_addr and npi_route_id</i>
	37	<i>2780/3780 Bisync Addresses</i>
	39	<i>Bisync Timing</i>
	39	hdl_T1 — Line Rebid Timeout
	39	hdl_T2 — Line Idle Timeout
	39	hdl_T3 — Transmit Timeout
	41	hdl_T5 — Receive Timeout
	41	hdl_T8 — Line Release Timeout
SECTION 4	43	<i>Configuring 3270 Bisync</i>
	45	Basic Configuration
	45	<i>Selecting Protocol Type</i>
	45	<i>Number Of Stations/Controller Units</i>
	45	<i>Number of Sessions</i>

47	<i>Configuring Transparent Mode or ASCII Transfer</i>
49	<i>Setting Up Routing</i>
49	<i>Important Parameters: npi_route_addr and npi_route_id</i>
51	<i>3270 Bisync Addresses</i>
53	<i>Bisync Timing</i>
53	hdl_T1 — Line Rebid Timeout
53	hdl_T3 — Transmit Timeout (not shown)
55	hdl_T5 — Receive Timeout (not shown)
55	hdl_T8 — Line Release Timeout (not shown)

SECTION 5

57	<i>Using the STREAMS Bisync API</i>
58	
59	Overview of the API
60	<i>ASCII or EBCDIC, Transparent Mode or Non-transparent Mode</i>
61	<i>Qualified Data</i>
62	A Selection of NPI API Routines
62	<i>Initialization and Bind</i>
62	<i>Connecting</i>
62	<i>Data Transfer</i>
62	<i>Disconnect and Close</i>

1

Configuring STREAMS Bisync

This chapter introduces Gcom's Bisync package. The components of the package are identified and the procedure of configuration outlined.

This chapter should be the starting point for all configurations.

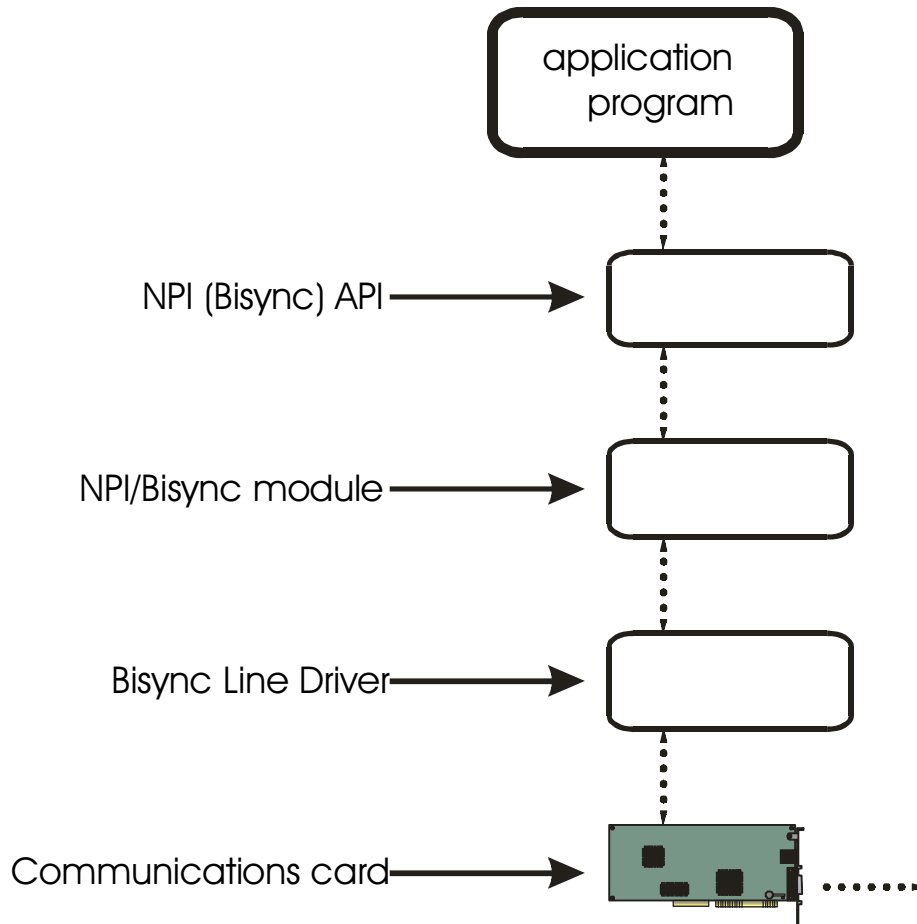


Figure 1 A STREAMS Bisync Stack

Components of the Bisync Stack

Three primary software components make up a GCOM STREAMS Bisync protocol stack: the line driver, the STREAMS Bisync module itself, and the NPI/Bisync 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 STREAMS Bisync module handles the actual Bisync protocol, maintaining appropriate states, handling protocol events, and managing addressing.

The NPI/Bisync API provides application programs access to the facilities the STREAMS Bisync module and the line driver offer.

An application program will use the NPI/Bisync API and the stack itself will employ the functionality of a synchronous communications card.

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13        # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1          # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0           # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz      = 8          # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user         = cdi
    hdl_pcl_kind  = 1          # 3780
    hdl_num_sesh  = 4
    hdl_num_sta   =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101       # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 2 A sample configuration file

The Configuration Process

Setting up a Bisync installation involves editing the configuration file used to start the *Gcom_monitor*. Several sample **configs** files can be found in the subdirectories of `/usr/lib/gcom/bisync`. The files are named with an extension appropriate to the protocol it's supposed to configure; **configs.3270** configures 3270 Bisync and **configs.2780** configures 2780 Bisync. Corresponding **start** scripts are also present in that directory. These scripts invoke *Gcom_monitor* in a way appropriate for that configuration.

Preparing a Bisync installation involves editing one of these files to match the local configuration.

Layout of the File

The file splits roughly in half, the first half configuring the line driver and the STREAMS facility, while the second half configures the Bisync module.

An installation will need to configure both the line driver (sometimes called the 'CDI module' or 'intsx25') and the Bisync module.

Chapter 2 starting on page 13 discusses configuration parameters for the line driver, while Chapter 3 starting on page 29 discusses configuring the Bisync module for 2780/3780 Bisync and Chapter 4 starting on page 43 discusses configuring the Bisync module for 3270.

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*. These parameters are not likely to 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 Line-level Bisync

This chapter identifies the line-level configuration file parameters which are likely to be of interest in a Bisync installation. Line-level configuration must be performed for every installation. Additional line-level parameters are identified and discussed in the *STREAMS Administrator's Guide*.

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10        # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13       # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1         # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0          # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz      = 8          # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user          = cdi
    hdl_pcl_kind  = 1          # 3780
    hdl_num_sesh  = 4
    hdl_num_sta   =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101        # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 3 A sample configuration file

Configuring the Line Driver

The *cdip* sections of the configuration file configure the line driver. To select settings for the line driver, locate the entry for *cdip.1* in a copy of your **configs** file. The *cdip.1* section should look similar to the one in the example to the left. Other sections may be different, and additional sections may be present, depending on the configuration file you select.

The two “normal” differences to expect are the presence of new parameters and the presence of multiple listings for the same parameter. These are discussed briefly on page 11.

A discussion of each of the other possible parameters follows.

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9         # Half-duplex Bisync ASCII sync
    ix25mode      = 10        # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13       # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1         # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0         # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz      = 8         # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27        # HD_line (bisync)
    provider      = npi
    user          = cdi
    hdl_pcl_kind  = 1         # 3780
    hdl_num_sesh  = 4
    hdl_num_sta   =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101      # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 4 A sample configuration file

***baudrate*—Baud Rate**

The *baudrate* parameter specifies the constant value to set the baud rate generator for the serial chip interface at the link. This constant is used in clock-generation mode, described in *ix25_clks—Clock Routing Options for NRZ* on page 25.

Baud rate is specified in bits per second. The *baudrate* parameter looks something like this:

```
baudrate          = 9600
```

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9         # Half-duplex Bisync ASCII sync
    ix25mode      = 10        # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13       # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1         # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz      = 0         # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz       = 8         # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user          = cdi
    hdl_pcl_kind  = 1         # 3780
    hdl_num_sesh  = 4
    hdl_num_sta   =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101       # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 5 A sample configuration file

***maxfrmsz*—Frame Size**

The *maxfrmsz* parameter sets the maximum size of a received frame in bytes, not including CRC bytes. In most applications, this parameter should be set to 0. This forces Bisync to calculate the maximum frame length based on the overall buffer size used by the GCOM drivers.

The overall buffer size is set in the *rsys.mod* section of the *configs* file, with the *bfrsize* parameter. This value should be set to encompass the largest expected frame. An additional “cushion” of 20 bytes beyond this should prove adequately roomy for internal addressing needs.

ix25mode

The *ix25mode* parameter specifies how the line driver should expect to communicate. For Bisync, the two possible values are:

<i>value</i>	<i>meaning</i>
9	Half-duplex Bisync (ASCII)
10	Half-duplex Bisync (EBCDIC)

Which setting a particular installation should use depends upon the way the target host expects to use the line.

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13        # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1          # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0           # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz      = 8          # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user          = cdi
    hdl_pcl_kind  = 1          # 3780
    hdl_num_sesh  = 4
    hdl_num_sta   =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101        # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 6 A sample configuration file

hdx_options

The *hdx_options* parameter specifies whether our line is full-duplex or not. It accepts three possible values:

<i>value</i>	<i>meaning</i>
0x00	Full-duplex data transfer
0x01	Half-duplex data transfer
0x03	Half-duplex data transfer, ignoring CTS.

Bisync will ordinarily use either 0x01 for standard operation or 0x03 when RTS must be asserted but CTS can safely be ignored. The 0x00 option is useful in cases where data transfer can be performed without using modem signals at all.

ix25_ll_options

ix25_ll_options is a collection of bits indicating option settings. Three options are meaningful for Bisync, and they may be ORed together to form a composite value.

<i>value</i>	<i>meaning</i>
0x08	This sets a large Bisync trace buffer. Recommended for all Bisync installations.
0x20	When <i>ix25mode</i> is 9 (ASCII Bisync), use odd parity
0x40	When <i>ix25mode</i> is 9 (ASCII Bisync), use even parity
0x80	When <i>ix25mode</i> is 9 (ASCII Bisync) and parity is being checked and generated, this option suppresses generation and checking of parity on SYN characters.

If neither 0x20 or 0x40 are set, no parity will be generated.

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13        # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1          # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0           # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz      = 8          # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user          = cdi
    hdl_pcl_kind  = 1          # 3780
    hdl_num_sesh  = 4
    hdl_num_sta   =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101       # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 7 A sample configuration file

seq_end_dly

The *seq_end_dly* parameter indicates the number of FF's to send at the end of a Bisync message. By default, this value is 1, and it generally shouldn't require modification.

lt_dly

lt_dly is the line turnaround delay. In Bisync, this translates to the number of syncs which should be sent before the beginning of a block. IBM's specification requires a value of at least 4 (the default) here. If the remote host seems to have trouble preparing to receive in time, increasing this count should give it a little more leeway.

ix25wrapflg

The *ix25wrapflg* parameter usually allows selection from a number of loopback methods. For Bisync, none of these loopback methods are allowed. The only legal value is 0.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 12
    n_upas       = 12
    n_lpas       = 8
npi.mod
    n_minors      = 200
    n_upas       = 200
    n_upas       = 64
    n_lpas       = 8
rsys.mod
    bfrsize      = 1960      # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.*
    ix25mode     = 9          # Half-duplex Bisync ASCII sync
    ix25mode     = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma      = 1
    rcv_dma      = 3
    ix25_chip_type = 13      # HD64570
    hertz        = 9830400
    lt_dly       = 4
    seq_end_dly  = 1
    ix25_ll_options = 0x08
    hdx_optns    = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks    = 1          # TxC output, RxC input
    baudrate     = 9600
    wd_timer     = 5000      # watchdog timer in milli-sec
    maxfrmsz    = 0          # calculate from bfrsize
    mdm_msk      = 0x03      # DTR and RTS
    frxcqsz     = 8          # nr xmit buffers on board
cdip.1
    client       = "npiu.1"
npiu.1
    module_type  = 27         # HD_line (bisync)
    provider     = npi
    user        = cdi
    hdl_pcl_kind = 1          # 3780
    hdl_num_sesh = 4
    hdl_num_sta  =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options  = 0x00      # no options set
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 = 101       # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 8 A sample configuration file

***ix25_clks*—Clock Routing Options for NRZ**

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 value can be used for NRZI and FM modes, but is not recommended. Currently, this value's behavior is identical to the value 1 when in NRZI and FM modes.
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. In addition, this is usually the clocking mode used for NRZI and FM modes. In these modes, the receive clock is always derived from the received data stream.
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.
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 dependent upon the existence of received data in order for the transmitter to derive its clocking.

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

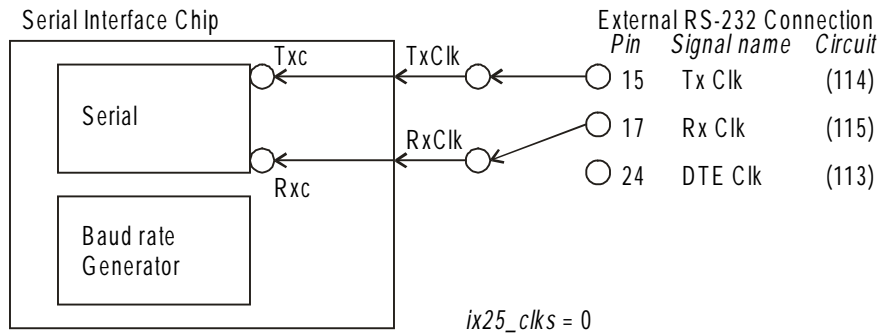


Figure 9 $ix25_clks = 0$

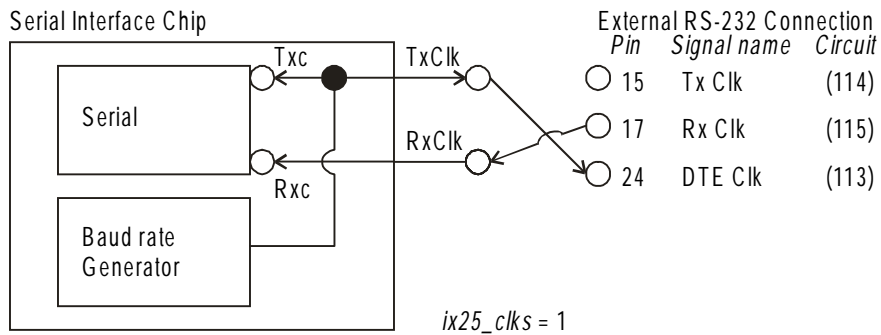


Figure 10 $ix25_clks = 1$

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.

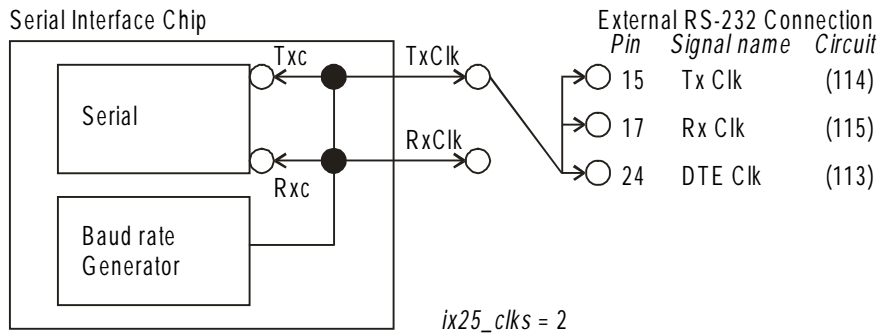


Figure 11 $ix25_clks = 2$

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

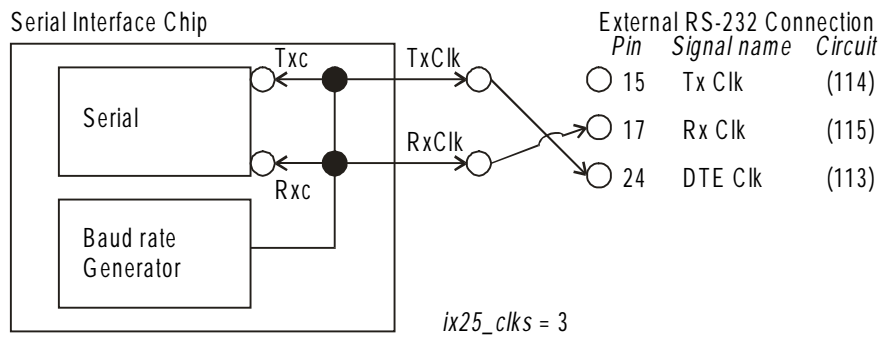


Figure 12 $ix25_clks = 3$

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.

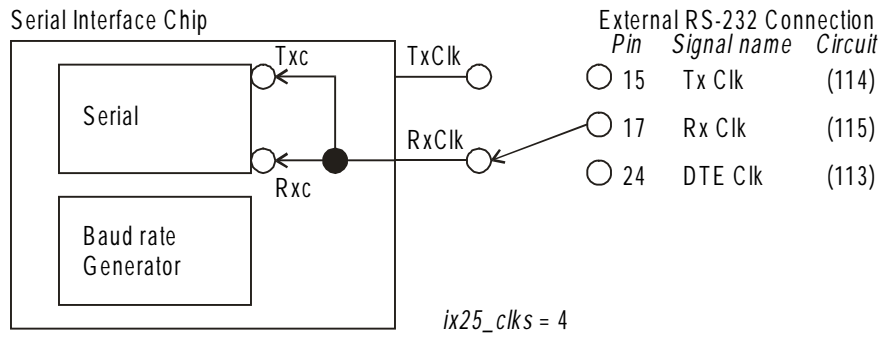


Figure 13 $ix25_clks = 4$

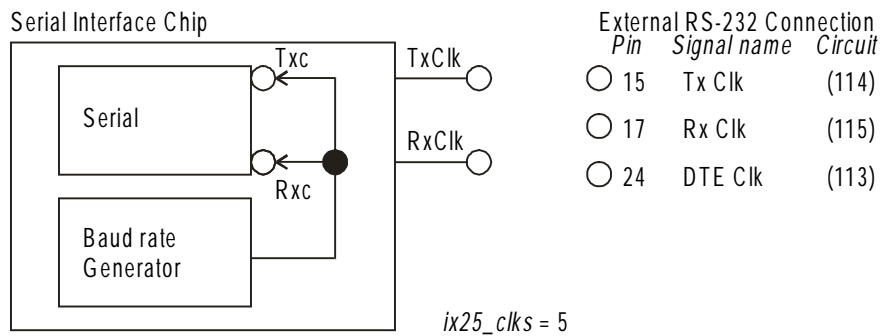


Figure 14 $ix25_clks = 5$

3

Configuring 2780/3780 Bisync

This chapter is designed to facilitate the configuration of 2780/3780 Bisync. It assumes that line-level configuration (covered in Chapter 2 starting on page 13) has already been performed, and covers the parameters which will need to be set in most installations. Additional parameters are discussed in the *STREAMS Administrator's Guide*.

```

cdi.mod
    n_minors      = 8
    n_upas        = 8
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13        # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1          # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0          # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz      = 8          # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user          = cdi
    hdl_pcl_kind  = 3          # 3780
    hdl_num_sesh  = 4
    hdl_trace_mask = 0xffff
    hdl_trace_mask = 0x0
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101       # unique id
    npi_route_addr = "99980301" # Bisync address

```

Figure 15 A sample configuration file

Basic Configuration

This section identifies the parameters which change in every configuration and explains how to determine what the parameter should be set to.

Selecting Protocol Type

The Gcom STREAMS Bisync module was designed to support several different Bisync versions. The *hdl_pcl_kind* parameter selects which version a particular line should emulate.

The sample configuration file at left uses '3', indicating a 2780/3780 device.

Table 2 Defines for *hdl_pcl_kind*

<i>Value</i>	<i>Description</i>
1	Gcom's 3270 Bisync Terminal Controller Emulator
2	Reserved for future use
3	Gcom's 2780/3780 Bisync Device Emulator

```

cdi.mod
    n_minors      = 8
    n_upas        = 8
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13        # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1          # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0           # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz      = 8          # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user         = cdi
    hdl_pcl_kind  = 3          # 3780
    hdl_num_sesh  = 4
    hdl_trace_mask = 0xffff
    hdl_trace_mask = 0x0
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101        # unique id
    npi_route_addr = "99980301" # Bisync address

```

Figure 16 A sample configuration file (repeated)

Configuring Transparent Mode and ASCII Transfer

hdl_options provides access to several optional configurations. Legal values and their meanings are listed in Table 3; multiple selections can be ORed together. For most of these, the default values will probably be appropriate. Some special cases include:

- 0x0001 Use ASCII instead of EBCDIC. Setting this bit allows use of ASCII across Bisync.
- 0x0002 Enable non-transparent transfer. Setting this bit turns transparent mode off. If transparent mode is turned off, buffers must not contain control characters.

Emulating a 2780/3780 Primary

When emulating a 2780/3780 primary, the 0x0004 bit of *hdl_options* should be set. If no other options are being employed, using 0x0004 for the value of *hdl_options* should produce the desired result.

The file at left is set up to be a 2780/3780 secondary, so this option is omitted.

Table 3 Values for *hdl_options*

<i>value</i>	<i>meaning</i>
0x0001	Use ASCII instead of EBCDIC
0x0002	Enable non-transparent transfer
0x0004	3780 Primary
0x0010	Permit use of RVI
0x0020	Permit use of limited conversational mode
0x0040	Do not send DLE-EOT
0x0080	Inhibit use of TTD
0x0100	Use SOH instead of STX

```

cdi.mod
    n_minors      = 8
    n_upas       = 8
    n_lpas       = 00
dlpi.mod
    n_minors      = 12
    n_upas       = 12
    n_lpas       = 8
npi.mod
    n_minors      = 200
    n_upas       = 200
    n_upas       = 64
    n_lpas       = 8
rsys.mod
    bfrsize      = 1960      # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.*
    ix25mode     = 9          # Half-duplex Bisync ASCII sync
    ix25mode     = 10        # Half-duplex Bisync EBCDIC sync
    xmt_dma      = 1
    rcv_dma      = 3
    ix25_chip_type = 13      # HD64570
    hertz        = 9830400
    lt_dly       = 4
    seq_end_dly  = 1
    ix25_ll_options = 0x08
    hdx_optns    = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks    = 1          # TxC output, RxC input
    baudrate     = 9600
    wd_timer     = 5000      # watchdog timer in milli-sec
    maxfrmsz    = 0          # calculate from bfrsize
    mdm_msk     = 0x03      # DTR and RTS
    frxcqsz     = 8          # nr xmit buffers on board
cdip.1
    client       = "npiu.1"
npiu.1
    module_type  = 27        # HD_line (bisync)
    provider     = npi
    user        = cdi
    hdl_pcl_kind = 3          # 3780
    hdl_num_sesh = 4
    hdl_trace_mask = 0xffff
    hdl_trace_mask = 0x0
    hdl_par_hist_sz = 166
    hdl_options  = 0x00      # no options set
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 = 101       # unique id
    npi_route_addr = "99980301" # Bisync address

```

Figure 17 A sample configuration file (repeated)

Setting Up Routing

Each Bisync module needs to have an entry in the NPI routing tables so that STREAMS knows how to properly direct traffic. This entry must be modified for nearly all installations.

The *npi.mn_ioctl.1* section in the configuration file at left presents an example of how NPI routing might be set up for a Bisync line. This example configures a single 3780 Bisync line.

Important Parameters: npi_route_addr and npi_route_id

Two options from the address-table entry impact Bisync operation. These two lines will be the address line (*npi_route_addr*) and the route ID (*npi_route_id*).

npi_route_id is an arbitrary number used as a way to identify this routing entry. It must be unique throughout the system, but may otherwise be any number.

npi_route_addr is a Bisync address indicating which addresses a particular Bisync module will attend to. These addresses can either be full addresses, or they can make use of wildcards to specify a group of addresses. Forming Bisync addresses is outlined below.

```

cdi.mod
    n_minors      = 8
    n_upas        = 8
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13        # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1          # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0          # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz       = 8          # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user          = cdi
    hdl_pcl_kind  = 3          # 3780
    hdl_num_sesh  = 4
    hdl_trace_mask = 0xffff
    hdl_trace_mask = 0x0
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101        # unique id
    npi_route_addr = "99980301" # Bisync address

```

Figure 18 A sample configuration file (repeated)

2780/3780 Bisync Addresses

2780/3780 Bisync addresses are strings composed of three components. The first component is a “signature” block of four characters that indicates the address is a Bisync address. This portion will always be 9998. The next two characters are the Bisync type, equivalent to the value from *hdl_pcl_kind*. The last two characters and the third block make up the line number.

While it’s not terribly useful for 2780/3780 Bisync, these addresses can include wildcards. A ? will match any single character, while a * will match any sequence of characters.

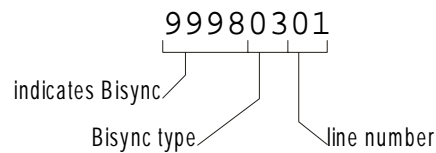


Figure 19 A 2780/3780 Address

- 9998 The first portion of the address is a tag indicating that the address is a Bisync address. This serves to prevent collisions with other addresses in the NPI routing table.
- 03 The next portion of the address is the Bisync type. The number used here is the same number used in *hdl_pcl_kind*. The '03' indicates 2780/3780 Bisync.
- 01 The third portion of the address is the line number. This is the physical line number. The line number is the same as the port number.

```

cdi.mod
    n_minors      = 8
    n_upas       = 8
    n_lpas       = 00
dlpi.mod
    n_minors      = 12
    n_upas       = 12
    n_lpas       = 8
npi.mod
    n_minors      = 200
    n_upas       = 200
    n_upas       = 64
    n_lpas       = 8
rsys.mod
    bfrsize      = 1960      # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.*
    ix25mode     = 9          # Half-duplex Bisync ASCII sync
    ix25mode     = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma      = 1
    rcv_dma      = 3
    ix25_chip_type = 13      # HD64570
    hertz        = 9830400
    lt_dly       = 4
    seq_end_dly  = 1
    ix25_ll_options = 0x08
    hdx_optns    = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks    = 1          # TxC output, RxC input
    baudrate     = 9600
    wd_timer     = 5000      # watchdog timer in milli-sec
    maxfrmsz    = 0          # calculate from bfrsize
    mdm_msk     = 0x03      # DTR and RTS
    frxcqsz     = 8          # nr xmit buffers on board
cdip.1
    client       = "npiu.1"
npiu.1
    module_type  = 27         # HD_line (bisync)
    provider     = npi
    user        = cdi
    hdl_pcl_kind = 3          # 3780
    hdl_num_sesh = 4
    hdl_trace_mask = 0xffff
    hdl_trace_mask = 0x0
    hdl_par_hist_sz = 166
    hdl_options  = 0x00      # no options set
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 = 101       # unique id
    npi_route_addr = "99980301" # Bisync address

```

Figure 20 A sample configuration file (repeated)

Bisync Timing

***hdl_T1* — Line Rebid Timeout**

In 2780/3780 Bisync, the *hdl_T1* usually is set to 1000 milliseconds (1 second) if the local machine will be the primary, or 3000 milliseconds (3 seconds) if the local machine will be the secondary, as in the sample file at left. The line rebid count, *hdl_K1*, limits the maximum number of times this timeout can be exceeded before clearing the call.

***hdl_T2* — Line Idle Timeout**

The *hdl_T2* timer is a simple idle timeout. If the timer expires before activity is detected on the line, the Bisync engine will bid for the line and yield it again immediately upon receiving a response.

***hdl_T3* — Transmit Timeout**

The transmit timeout, *hdl_T3*, limits the amount of time allowed to complete a transmission before aborting the transmission. The *hdl_K3* count is the maximum number of times this can occur before clearing the call.

Table 4 2780/3780 Timers

<i>parameter</i>	<i>default</i>	<i>legal values</i>	<i>description</i>
<i>hdl_T1</i>	1000	0-65000	Line rebid timeout
<i>hdl_K1</i>	3	0-65000	Line rebid count
<i>hdl_T2</i>	15000	0-65000	Line idle timeout
<i>hdl_T3</i>	1000	0-65000	Transmit timeout
<i>hdl_K3</i>	3	0-65000	Transmit timeout count
<i>hdl_T5</i>	3000	0-65000	Receive timeout
<i>hdl_K5</i>	5	0-65000	Receive timeout count
<i>hdl_T8</i>	500	0-65000	Line yield pause
<i>hdl_K8</i>	0	0-65000	Line yield count

```

cdi.mod
    n_minors      = 8
    n_upas       = 8
    n_lpas       = 00
dlpi.mod
    n_minors      = 12
    n_upas       = 12
    n_lpas       = 8
npi.mod
    n_minors      = 200
    n_upas       = 200
    n_upas       = 64
    n_lpas       = 8
rsys.mod
    bfrsize      = 1960      # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.*
    ix25mode     = 9          # Half-duplex Bisync ASCII sync
    ix25mode     = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma      = 1
    rcv_dma      = 3
    ix25_chip_type = 13      # HD64570
    hertz        = 9830400
    lt_dly       = 4
    seq_end_dly  = 1
    ix25_ll_options = 0x08
    hdx_optns    = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks    = 1          # TxC output, RxC input
    baudrate     = 9600
    wd_timer     = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0          # calculate from bfrsize
    mdm_msk      = 0x03      # DTR and RTS
    frxcqsz     = 8          # nr xmit buffers on board
cdip.1
    client       = "npiu.1"
npiu.1
    module_type  = 27         # HD_line (bisync)
    provider     = npi
    user         = cdi
    hdl_pcl_kind = 3          # 3780
    hdl_num_sesh = 4
    hdl_trace_mask = 0xffff
    hdl_trace_mask = 0x0
    hdl_par_hist_sz = 166
    hdl_options  = 0x00      # no options set
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 = 101      # unique id
    npi_route_addr = "99980301" # Bisync address

```

Figure 21 A sample configuration file (repeated)

***hdl_T5* — Receive Timeout**

The *hdl_T5* receive timeout is likewise the maximum amount of time allowed for completion of a receive. The *hdl_K5* limits the maximum number of times this timeout can be exceeded before clearing the call.

***hdl_T8* — Line Release Timeout**

The *hdl_T8* and *hdl_K8* are slightly different. The *hdl_K8* is the maximum number of bytes to transmit before releasing control of the line to allow the other end to transmit. The *hdl_T8* timer indicates how long the Bisync engine should then wait before bidding for control of the line again. If *hdl_K8* is zero, this mechanism is bypassed.

Table 5 2780/3780 Timers (repeated)

<i>parameter</i>	<i>default</i>	<i>legal values</i>	<i>description</i>
<i>hdl_T1</i>	1000	0-65000	Line rebid timeout
<i>hdl_K1</i>	3	0-65000	Line rebid count
<i>hdl_T2</i>	15000	0-65000	Line idle timeout
<i>hdl_T3</i>	1000	0-65000	Transmit timeout
<i>hdl_K3</i>	3	0-65000	Transmit timeout count
<i>hdl_T5</i>	3000	0-65000	Receive timeout
<i>hdl_K5</i>	5	0-65000	Receive timeout count
<i>hdl_T8</i>	500	0-65000	Line yield pause
<i>hdl_K8</i>	0	0-65000	Line yield count

4

Configuring 3270 Bisync

This chapter is designed to facilitate the configuration of 3270 Bisync. It assumes that line-level configuration (covered in Chapter 2 starting on page 13) has already been performed, and covers the parameters which will need to be set in most installations. Additional parameters are discussed in the *STREAMS Administrator's Guide*.

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10        # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13       # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1         # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0          # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz      = 8          # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user          = cdi
    hdl_pcl_kind  = 1          # 3780
    hdl_num_sesh  = 4
    hdl_num_sta   =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101       # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 22 A sample configuration file

Basic Configuration

This section identifies the parameters which change in every configuration and explains how to determine what the parameter should be set to.

Selecting Protocol Type

The Gcom STREAMS Bisync module was designed to support several different Bisync versions. The *hdl_pcl_kind* parameter selects which version a particular line should emulate.

The sample configuration file at left uses '1', indicating a 3270 device.

Table 6 Defines for *hdl_pcl_kind*

<i>Value</i>	<i>Description</i>
1	Gcom's 3270 Bisync Terminal Controller Emulator
2	Reserved for future use
3	Gcom's 2780/3780 Bisync Device Emulator

Number Of Stations/Controller Units

hdl_num_sta configures the number of stations in the station table. This must be equal to the number of devices plus the number of emulated controller units.

Number of Sessions

hdl_num_sesh is the number of sessions in the station table. This must be equal to the number of stations (*hdl_num_sta*) plus 1.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 12
    n_upas       = 12
    n_lpas       = 8
npi.mod
    n_minors      = 200
    n_upas       = 200
    n_upas       = 64
    n_lpas       = 8
rsys.mod
    bfrsize      = 1960      # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.*
    ix25mode     = 9          # Half-duplex Bisync ASCII sync
    ix25mode     = 10        # Half-duplex Bisync EBCDIC sync
    xmt_dma      = 1
    rcv_dma      = 3
    ix25_chip_type = 13      # HD64570
    hertz        = 9830400
    lt_dly       = 4
    seq_end_dly  = 1
    ix25_ll_options = 0x08
    hdx_optns    = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks    = 1          # TxC output, RxC input
    baudrate     = 9600
    wd_timer     = 5000      # watchdog timer in milli-sec
    maxfrmsz    = 0          # calculate from bfrsize
    mdm_msk     = 0x03      # DTR and RTS
    frxcqsz     = 8          # nr xmit buffers on board
cdip.1
    client       = "npiu.1"
npiu.1
    module_type  = 27          # HD_line (bisync)
    provider     = npi
    user        = cdi
    hdl_pcl_kind = 1          # 3780
    hdl_num_sesh = 4
    hdl_num_sta  =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options  = 0x00      # no options set
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 = 101        # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 23 A sample configuration file (repeated)

Configuring Transparent Mode or ASCII Transfer

hdl_options provides access to several optional configuration parameters. Legal values and their meanings are listed in Table 7; multiple selections can be ORed together. For most of these, the default values will probably be appropriate.

Some special cases where default may not be appropriate include:

- 0x0001 ASCII mode. Setting this bit allows use of ASCII characters in Bisync communication.
- 0x0002 Non-transparent mode. Setting this bit turns transparent mode off. If transparent mode is turned off, buffers must not contain control characters.

Table 7 Values for *hdl_options*

<i>value</i>	<i>meaning</i>
0x0001	Use ASCII instead of EBCDIC
0x0002	Enable non-transparent transfer
0x0004	3780 Primary
0x0010	Permit use of RVI
0x0020	Permit use of limited conversational mode
0x0040	Do not send DLE-EOT
0x0080	Inhibit use of TTD
0x0100	Use SOH instead of STX

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13        # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1         # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz      = 0         # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz       = 8         # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user         = cdi
    hdl_pcl_kind  = 1          # 3780
    hdl_num_sesh  = 4
    hdl_num_sta   =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101        # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 24 A sample configuration file (repeated)

Setting Up Routing

Each Bisync module needs to have an entry in the NPI routing tables so that STREAMS knows how to properly direct traffic. This entry must be modified for nearly all installations.

The *npi.mn_ioctl.1* section in the configuration file at left presents an example of how NPI routing might be set up for a Bisync line. This example configures a single 3270 Bisync line.

Important Parameters: npi_route_addr and npi_route_id

Two options from the address-table entry impact Bisync operation. These two lines will be the address line (*npi_route_addr*) and the route ID (*npi_route_id*).

npi_route_id is an arbitrary number used as a way to identify this routing entry. It must be unique throughout the system, but may otherwise be any number.

npi_route_addr is a Bisync address indicating which addresses a particular Bisync module will attend to. These addresses can either be full addresses, or they can make use of wildcards to specify a group of addresses. Forming Bisync addresses is outlined below.

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13        # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1          # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0          # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz      = 8          # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user          = cdi
    hdl_pcl_kind  = 1          # 3780
    hdl_num_sesh  = 4
    hdl_num_sta   =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101       # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 25 A sample configuration file (repeated)

3270 Bisync Addresses

3270 Bisync addresses are simple strings composed of five components. The first component is a “signature” block of four characters that indicates the address is a Bisync address. This portion will always be 9998. The next two characters are the Bisync type, equivalent to the value from *hdl_pcl_kind*. The next two characters (and the third block) represent up the line number. The next three characters are the controller unit address and the last three characters are the station address. Both the controller unit and the station addresses are expressed in decimal (rather than the usual hexadecimal).

These Bisync addresses can include wildcards. A ? will match any single character, while a * will match any sequence of characters. This could be useful when preparing a single application that will, for example, manage several Bisync stations.

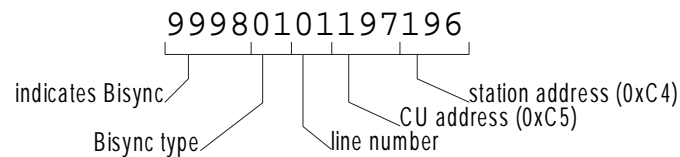


Figure 26 A 3270 Address

- 9998 The first portion of the address is a tag indicating that the address is a Bisync address. This serves to prevent collisions with other addresses in the NPI routing table.
- 01 The second portion of the address is the Bisync type. The number used here is the same number used in *hdl_pcl_kind*. The ‘01’ indicates that this is a 3270 Controller Unit emulator.
- 01 The next portion is the line number. The line number is the same as the port number.
- 197 The fourth portion is the controller unit address. In the example, this is 0xC5 expressed as a decimal number (197). Each line can support up to 32 emulated controller units.
- 196 The final portion of the address is the station address. In the example, this is 0xC4 expressed as a decimal number (196). Each controller unit can support up to 32 emulated stations.

```

cdi.mod
    n_minors      = 2
    n_upas       = 2
    n_lpas       = 00
dlpi.mod
    n_minors      = 12
    n_upas       = 12
    n_lpas       = 8
npi.mod
    n_minors      = 200
    n_upas       = 200
    n_upas       = 64
    n_lpas       = 8
rsys.mod
    bfrsize      = 1960      # large buffer data space
    r_unix_ticks = 5
    r_timeout_ms = 50
cdip.*
    ix25mode     = 9          # Half-duplex Bisync ASCII sync
    ix25mode     = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma      = 1
    rcv_dma      = 3
    ix25_chip_type = 13      # HD64570
    hertz        = 9830400
    lt_dly       = 4
    seq_end_dly  = 1
    ix25_ll_options = 0x08
    hdx_optns    = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks    = 1          # TxC output, RxC input
    baudrate     = 9600
    wd_timer     = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0          # calculate from bfrsize
    mdm_msk      = 0x03      # DTR and RTS
    frxcqsz     = 8          # nr xmit buffers on board
cdip.1
    client       = "npiu.1"
npiu.1
    module_type  = 27         # HD_line (bisync)
    provider     = npi
    user        = cdi
    hdl_pcl_kind = 1          # 3780
    hdl_num_sesh = 4
    hdl_num_sta  =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options  = 0x00      # no options set
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 = 101       # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 27 A sample configuration file (repeated)

Bisync Timing

***hdl_T1* — Line Rebid Timeout**

In 3270 Bisync, the *hdl_T1* usually is set to 1000 milliseconds (1 second), which is the default. The line rebid count, *hdl_K1*, limits the maximum number of times this timeout can be exceeded before clearing the call.

***hdl_T3* — Transmit Timeout (not shown)**

The transmit timeout, *hdl_T3*, limits the amount of time allowed to complete a transmission before aborting the transmission. The *hdl_K3* count is the maximum number of times this can occur before clearing the call.

Table 8 3270 Timers

<i>parameter</i>	<i>default</i>	<i>legal values</i>	<i>description</i>
<i>hdl_T1</i>	1000	0-65000	Line rebid timeout
<i>hdl_K1</i>	3	0-65000	Line rebid count
<i>hdl_T2</i>	15000	0-65000	Line idle timeout
<i>hdl_T3</i>	1000	0-65000	Transmit timeout
<i>hdl_K3</i>	3	0-65000	Transmit timeout count
<i>hdl_T5</i>	3000	0-65000	Receive timeout
<i>hdl_K5</i>	5	0-65000	Receive timeout count
<i>hdl_T8</i>	500	0-65000	Line yield pause
<i>hdl_K8</i>	0	0-65000	Line yield count

```

cdi.mod
    n_minors      = 2
    n_upas        = 2
    n_lpas        = 00
dlpi.mod
    n_minors      = 12
    n_upas        = 12
    n_lpas        = 8
npi.mod
    n_minors      = 200
    n_upas        = 200
    n_upas        = 64
    n_lpas        = 8
rsys.mod
    bfrsize       = 1960      # large buffer data space
    r_unix_ticks  = 5
    r_timeout_ms  = 50
cdip.*
    ix25mode      = 9          # Half-duplex Bisync ASCII sync
    ix25mode      = 10         # Half-duplex Bisync EBCDIC sync
    xmt_dma       = 1
    rcv_dma       = 3
    ix25_chip_type = 13        # HD64570
    hertz         = 9830400
    lt_dly        = 4
    seq_end_dly   = 1
    ix25_ll_options = 0x08
    hdx_optns     = 0x03      # 01=HDX, 02=no CTS wait
    ix25_clks     = 1         # TxC output, RxC input
    baudrate      = 9600
    wd_timer      = 5000      # watchdog timer in milli-sec
    maxfrmsz     = 0          # calculate from bfrsize
    mdm_msk       = 0x03      # DTR and RTS
    frxcqsz      = 8          # nr xmit buffers on board
cdip.1
    client        = "npiu.1"
npiu.1
    module_type   = 27         # HD_line (bisync)
    provider      = npi
    user          = cdi
    hdl_pcl_kind  = 1          # 3780
    hdl_num_sesh  = 4
    hdl_num_sta   =
    hdl_trace_mask = 0xffff
    hdl_par_hist_sz = 166
    hdl_options   = 0x00      # no options set
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  = 101       # unique id
    npi_route_addr = "99980101197196" # Bisync address

```

Figure 28 A sample configuration file (repeated)

***hdl_T5* — Receive Timeout (not shown)**

The *hdl_T5* receive timeout is likewise the maximum amount of time allowed for completion of a receive. The *hdl_K5* limits the maximum number of times this timeout can be exceeded before clearing the call.

***hdl_T8* — Line Release Timeout (not shown)**

The *hdl_T8* and *hdl_K8* are slightly different. The *hdl_K8* is the maximum number of blocks to transmit before releasing control of the line. The *hdl_T8* timer is unused in 3270 Bisync. If *hdl_K8* is zero, this mechanism is bypassed.

Table 9 2780/3780 Timers (repeated)

<i>parameter</i>	<i>default</i>	<i>legal values</i>	<i>description</i>
<i>hdl_T1</i>	1000	0-65000	Line rebid timeout
<i>hdl_K1</i>	3	0-65000	Line rebid count
<i>hdl_T2</i>	15000	0-65000	Line idle timeout
<i>hdl_T3</i>	1000	0-65000	Transmit timeout
<i>hdl_K3</i>	3	0-65000	Transmit timeout count
<i>hdl_T5</i>	3000	0-65000	Receive timeout
<i>hdl_K5</i>	5	0-65000	Receive timeout count
<i>hdl_T8</i>	500	0-65000	Line yield pause
<i>hdl_K8</i>	0	0-65000	Line yield count

5

Using the STREAMS Bisync API

This chapter provides a quick overview of the way Bisync 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 *STREAMS Bisync API Guide* and the *NPI API Guide* for additional information on the use of the API or the functions it presents.

Overview of the API

The STREAMS Bisync API uses the NPI API to provide access to Bisync features. Normal sending and receiving of regular data are accomplished just as for any other NPI protocol. Bisync framing characters are not a part of the NPI data stream.

Applications begin by initializing the API's data space and opening a logfile. They open a data stream, then bind that stream to an address. Applications will then connect to the remote host using a standard Bisync address (see "3270 Bisync Addresses" on page 51 or "2780/3780 Bisync Addresses" on page 37).

ASCII or EBCDIC, Transparent Mode or Non-transparent Mode

Once a connection has been established, it will be ready for data transfer. Data transfer options are set up in the configuration file as part of the Bisync module. These options are described on page 47 and page 33. Options in the *hdl_options* parameter control whether Bisync will use ASCII or EBCDIC and whether data transfer will be transparent-mode or nontransparent-mode.

Qualified Data

During data transfer, the API provides Bisync protocol control data through the qualified data mechanism of the NPI API. This provides easy separation of control information from normal data.

When receiving data, buffers containing control data will be flagged as qualified data. These buffers will begin with either an 0x07 byte or an 0x05 byte. Buffers beginning with 0x07 will contain an additional byte of protocol control data. Buffers beginning with 0x05 will contain an additional byte of error information. These buffers may also contain additional meaningful information beyond the error code, depending on circumstances.

Table 10 Format of Qualified Data Buffers

<i>Byte 0</i>	<i>Byte 1</i>	<i>Description</i>
0x05	Error code	For errors. May also have additional error information beyond 2 bytes.
0x07	Protocol message	For protocol control messages. Will be 2 bytes in length.

A Selection of NPI API Routines

Initialization and Bind

<i>npi_init()</i>	initializes the API's data structures and opens a logfile
<i>npi_open_data()</i>	opens a data stream
<i>npi_bind_ascii_nsap()</i>	binds the data stream to an address pattern

Connecting

<i>npi_connect()</i>	opens a data stream, binds it, and initiates a connection on it
<i>npi_connect_req()</i>	initiates a connection on a previously opened and bound data stream
<i>npi_send_connect_req()</i>	initiates a connection but does not wait for the connection to be established before returning

Data Transfer

<i>npi_read_data()</i>	simple read routine for managing data transfer
<i>npi_rcv()</i>	a more complex read routine with provision for data flags
<i>npi_write_data()</i>	simple write routine designed to write normal data
<i>npi_put_data_proto()</i>	more complex write routine designed to allow use of flags with the data

Disconnect and Close

<i>npi_discon_req()</i>	gracefully shuts down a Bisync connection
<i>close()</i>	closes the stream and releases its resources