http://www.bintec.de

RELEASE NOTE
BIANCA/BRICK-XL

July 24, 1998

New System Software:
Release 4.8 Revision 6

This document describes the new features, enhancements, bug-fixes, and changes to the BIANCA / BRICK-XL System Software since Release 4.8 Revision 1.

NEW

Upgrading System Software

1. Retrieve the current system software image from BinTec's WWW server at http://www.bintec.de.
2. With this image you can upgrade the BIANCA/BRICK-XL with the **update** command from the SNMP shell via a remote host (i.e. using telnet, minipad, or isdnlogin) or by using the **BOOTmonitor** if you are logged in directly on the console. Information on using the BOOTmonitor can be found in the *BRICK-XL User's Guide* under *Firmware Upgrades*.
3. Once you've installed Release 4.8 Revision 6 you may want to retrieve the latest documentation (in Adobe's PDF format), which is also available from BinTec's FTP server at the address noted above.

   Note: When upgrading system software, it is also recommended that you use the most current versions of *BRICKware for Windows* and *UNIXTools*. Both can be retrieved from BinTec's FTP server.

What's New in Release 4.8.6

Release 4.8 Revision 6:

Released: 24.07.98

Features: Bugfixes: Detailed Description:

Known Problems

In Release 4.8 Revision 6, three problems currently exist.

1. **Network Address Translation**
   After receiving several broadcast packets via an interface where NAT is being performed the BRICK may either "lock-up" or inadvertently reboot. If the system locks up the BRICK will no longer be accessible (via remote or console) and must be power cycled on and off.
2. **Dial-up connections for RADIUS-Users**
   Interfaces configured to use `ip_lapb` encapsulation, using the following entry in /etc/raddb/users,
   `BinTec-biboPPPTable = "Encapsulation=ip_lapb"`
   are sometimes rejected by the BRICK.
3. **V.90 Modem Connections**
   Depending on local line quality, V.90 modem connections on the BRICK may inadvertently disconnect, or may not be possible.

Features

Microsoft Callback Extension to Mode 3

The Microsoft Callback Control Protocol (CBCP) knows different modes to decide among other things, which number is used for callback.

Up to now Mode 2 was implemented. In Mode 2 (callback to a user-defined number) the user is asked, when calling from a Windows95/ NT client, to enter the callback number. This number is then used for callback.

From this release on the MS-CBCP was extended to Mode 3. Mode 3 uses a predefined number for callback.

Which mode is used (Mode 2 or Mode 3) depends on whether there is a predefined number assigned. When there is a predefined number, either a entry in the *biboPPPDialTable* for this partner (direction both or outgoing) or when authentication is made via RADIUS the variable Callback-Number is assigned, then Mode 3 is used. With no number assigned callback is made using Mode 2.

Such it is ensured that a callback is either made using the user-specified or the predefined number.

The variable *CallBack* in the *biboPPPTable* can be set to *ppp_offered* or *enabled*. But you must notice that with the value set to *enabled* no authentication is made during callback.

With the variable *CallBack* in the *biboPPPTable* assigned the value *ppp_offered* and the *biboPPPDialTable* containing the entry *isdn* for *Type* and a *Number* specified, the callback is made using Mode 3 for a call from a Windows95/ NT client identified inband (via the internal *biboPPPTable*) or outband (via CLID). When the values in the *biboPPPDialTable* are not specified, Mode 2 is used.

With the variable *CallBack* in the *biboPPPTable* assigned the value *enabled* and the initial call is identified inband and without RADIUS the settings are the same as described above for *ppp_offered*.

Also see Microsoft Callback via RADIUS below.

Microsoft Callback via RADIUS

With this new release it is possible to use Microsoft Callback via RADIUS for calls from a Windows95/ NT client.

The RADIUS server must be configured as follows:

```
Service-Type = Callback-Framed
```

Specifying only the Service-Type means using Mode 2 of the CBCP (user-specifiable number). This configuration assigns the value *enabled* to the variable *biboPPPCallBack*.

To use Mode 3 (predefined number), that means using a fixed callback number, you must additionally assign a callback number as in the following example:

Service-Type = Callback-Framed

Callback-Number = “392”

The feature Microsoft Callback via RADIUS is only available for an inband identification of the caller (no calling line identification). The same it’s not possible to set the value *ppp_offered* for the variable *biboPPPCallBack*. For the time of the PPP connection there exists a temporary entry in the *biboPPPTable* with variable *CallBack* assigned the value *enabled*. This means that there is no additional authentication during callback. In Mode 3 a temporary entry in the *biboPPPDialTable* using the defined calling number is generated, too.

IPX RADIUS Extensions

The BRICK now supports dial-up IPX client connections via RADIUS. For a detailed description of this new feature see IPX RADIUS Extensions on page 17.

Frame Relay

Beginning with Release 8 Revision 6 Frame Relay is officially supported on the BRICK-XL. The BRICK-XL can be used as a Frame Relay Switch or a Frame Relay Router and supports the following official and defacto standards:

RFC 1490 *Multiprotocol Interconnect over Frame Relay*
RFC 1293 *Inverse Address Resolution Protocol*
ITU-T Q933a, Appendix II, X6 *Line Management Extensions*
FRF 1.1 *Congestion Management*

Frame Relay requires a separate license to be installed on the BRICK and may be purchased directly from BinTec Communications or your local distributor.

A detailed description of Frame Relay on the BRICK (Setup Tool menus and SNMP shell MIB objects/tables) as well as background information for users not familiar with Frame Relay is contained in the section Frame Relay beginning on page 20.

X.25

X.25 Window/ Packet Size Negotiation

Now you can decide for each X.25 link, whether a window/ packet size negotiation is made.

x25LkPrNegotiation is the new parameter in the *x25LinkPresetTable*, which handles this feature. This parameter can be assigned three possible values:

never	No negotiation. When a call arrives that does not correspond to the default size, the call is cleared.
always	Negotiations are always made.
when_necessary	There are only negotiations, when the requested values differ from the default values.

Window/ packet size negotiation settings can also be configured via Setup Tool, see “Configuring X.25 Parameters in Setup Tool” on page 6.

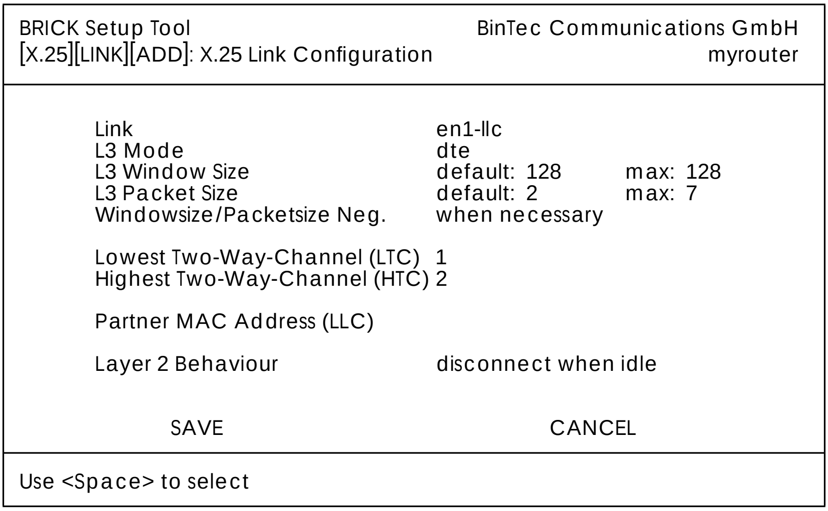
Configuring X.25 Parameters in Setup Tool

Now it is possible to configure additional X.25 parameters using Setup Tool.

X.25 Link Configuration

X.25 → **LINK CONFIGURATION** → **EDIT**

Here window/ packet size negotiation can be adjusted for an X.25 link. **Windowsize/ Packetsize Neg.** corresponds to the parameter *x25LkPrNegotiation* in the *x25LinkPresetTable*.



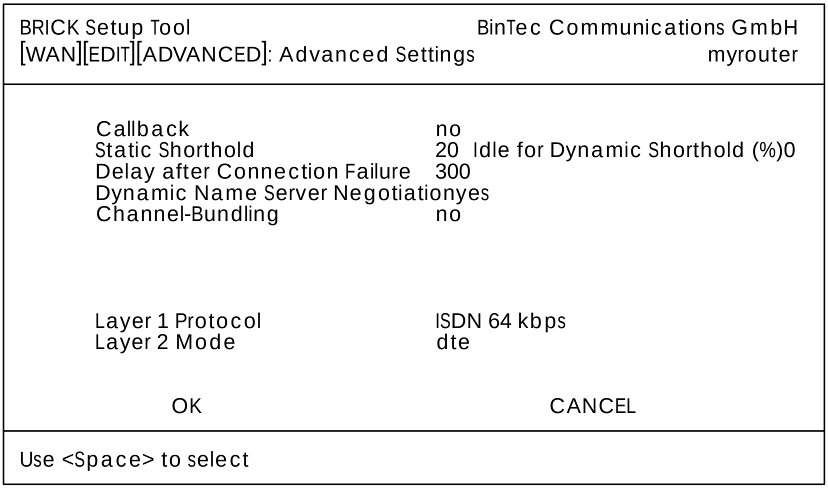
```
BRICK Setup Tool                                  BinTec Communications GmbH
[X.25][LINK][ADD]: X.25 Link Configuration                          myrouter

       Link                              en1-llc
       L3 Mode                           dte
       L3 Window Size                    default:  128      max:  128
       L3 Packet Size                    default:  2        max:  7
       Windowsize/Packetsize Neg.        when necessary

       Lowest Two-Way-Channel (LTC)  1
       Highest Two-Way-Channel (HTC) 2

       Partner MAC Address (LLC)

       Layer 2 Behaviour                 disconnect when idle


             SAVE                                  CANCEL

Use <Space> to select
```

Windowsize/ Packetsize Neg. = Decides whether window/ packetsize negotiation is made for this X.25 link. The possible values are **never**, **always** and **when necessary**, where **when necessary** is the default value. The value *never* means no negotiation. When a call arrives that does not correspond to the default size, the call is cleared. *Always* means negotiations are always made and when *when necessary* is selected, there are only negotiations, when the requested values differ from the default values.

WAN Partner

WAN PARTNER → **ADD** → **ADVANCED SETTINGS**

For WAN partners using the protocols X.25, X25ppp, X.31 B-Channel or X.25 no signalling the Layer 2 Mode can be configured in the advanced settings. The item Layer 2 Mode corresponds to the parameter *biboPPPLayer2Mode* in the *biboPPPTable.*



```
BRICK Setup Tool                              BinTec Communications GmbH
[WAN][EDIT][ADVANCED]: Advanced Settings                        myrouter

        Callback                          no
        Static Shorthold                  20  Idle for Dynamic Shorthold (%)0
        Delay after Connection Failure    300
        Dynamic Name Server Negotiationyes
        Channel-Bundling                  no



        Layer 1 Protocol                  ISDN 64 kbps
        Layer 2 Mode                      dte

               OK                                   CANCEL

Use <Space> to select
```

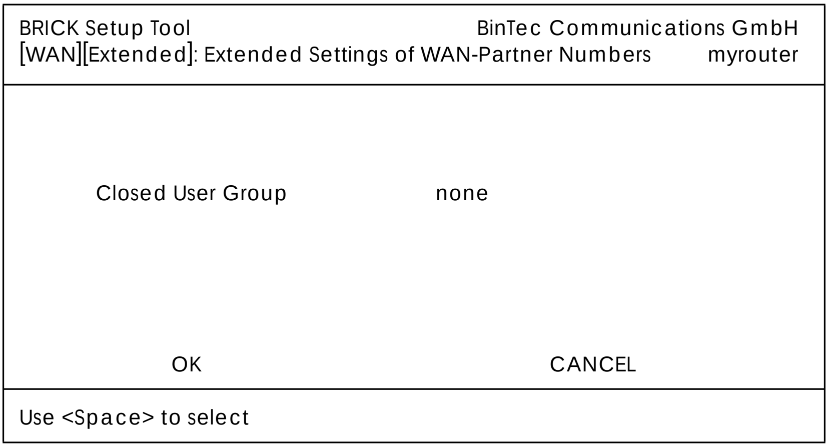
Layer 2 Mode = Layer 2 Mode can receive the values **auto**, **dte** or **dce**, where **auto** is the default value

WAN Partner Numbers Extended

WAN PARTNER → **ADD** → **WAN NUMBERS** →
→ **ADD** → **ADVANCED SETTINGS** →

Here the item Closed User Group can be configured. The item corresponds to the parameter *biboDialClosedUserGroup* in the *biboDialTable*

.



```
BRICK Setup Tool                                BinTec Communications GmbH
[WAN][Extended]: Extended Settings of WAN-Partner Numbers         myrouter



      Closed User Group                 none




            OK                               CANCEL

Use <Space> to select
```

Closed User Group = The item Closed User Group can be assigned the values **none** or an integer from **1 to 9999**. **None** is the default value.

Active Layer 2 Set Up for Incoming X.25 Calls

Prior to release 4.8.6 the BRICK remained passive during setup of incoming X.25 dialup connections and waited for a SABM (Set Asynchronous Balance Mode) from the caller. Some X.25 implementations however were also waiting for a SABM from the BRICK.

Now the BRICK only waits one second for an incoming SABM.

If no SABM is received within this time, the BRICK will send a SABM.

Because of the wait time the probability of a layer 2 setup collision is very small. Standard end-devices handle such collision correctly.

TP0 Bridge Extensions

To make possible connections between TCP clients and an X.25 network the TP0 Bridge feature (RFC 1086 / RFC 1006) is implemented on the BRICK.

With this release two extensions concerning the transmission of X.25 data (RFC 1006) for incoming X.25 connections to the TP0 Bridge have been made.

Firstly with release 4.8.6 NSAP addresses, which are subaddresses of X.25 addresses, can be proofed for incoming X.25 calls. If a listener transfers a NSAP address in the facility field of the listening address, only X.25 calls with the same NSAP address are signaled to the listener.

The second extension concerns the X.25 call indication packet, which is sent as the first packet, when an incoming X.25 connection is etablished. Now with release 4.8.6 there is a possibility that the listening application gets some information about the contents of the X.25 call indication packet.

To get this data the value of the function byte, (the first byte, the listener sends to the TP0 brigde, see RFC 1086) has to be 66 instead of 2. Then the first data packet, the listener receives on its new established TCP stream has the following format: It consists of 4 Byte TP0 header and the data in the extended X.25 Address Format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| vrsn = 255    |  reserved = 3 |          packet length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        address  type  = 4     | X.25 addr len | X.25 address  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      ...      |      ...      |      ...      |      ...      |
|      ...      |      ...      |      ...      |      ...      |
|      ...      |      ...      |      ...      |      ...      |
|      ...      |      ...      |      ...      |Protocol ID len|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Protocol ID  |      ...      |      ...      |      ...      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| CUDF length   | Call User Data field ...      |      ...      |
|      ...      |      ...      |      ...      |      ...      |
|      ...      |      ...      |      ...      |      ...      |
|      ...      |      ...      |      ...      |      ...      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Facility Length|  X.25 Facilities     ...      |      ...      |
|      ...      |      ...      |      ...      |      ...      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

V.90 for BIANCA/BRICK XL and XMP

The new ITU-T standard V.90 has been implemented in the K56flex modem features of the FM-8MOD Function Module for BRICK XL and BRICK XMP. These BRICKs now support the V.90 code and K56flex.

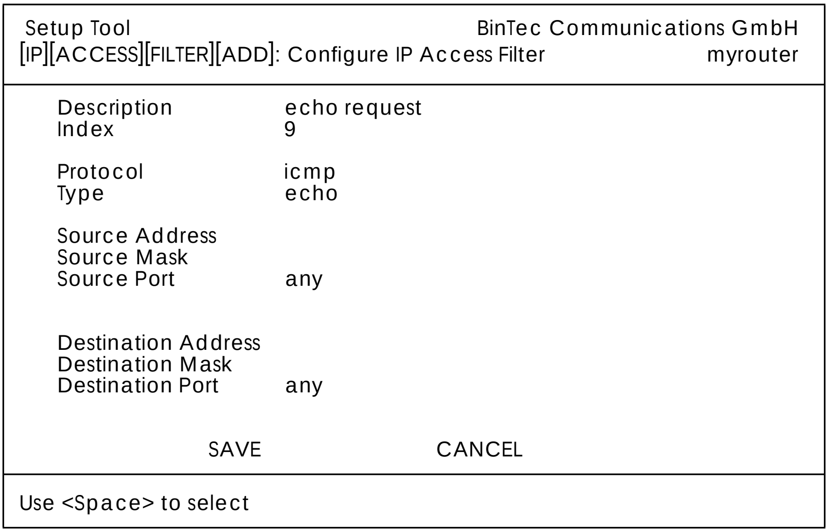
IP Filter for TCP State and ICMP Type

The filters for IP access have been enhanced.

ICMP Type

The filters can now be used to filter IP packets in dependence of the ICMP type.
In the *ipFiltertable* there is the new variable *icmptype*, which can be assigned the following values :
echoRep, *destUnreach*, *srcQuench*, *redirect*, *echo*, *timeExcds*, *parmProb*, *timestamp*, *timestampRep*, *addrMask*, *addr-MaskRep*, *dont_verify* .
Setup Tool's Filters menu has also been changed. You can now define filters according to appropriate ICMP types using the Type field after setting the protocol field to "ICMP".

IP → ACCESS LISTS → FILTERS → ADD


```
 Setup Tool                                      BinTec Communications GmbH
[IP][ACCESS][FILTER][ADD]: Configure IP Access Filter              myrouter

    Description              echo request
    Index                    9

    Protocol                 icmp
    Type                     echo

    Source Address
    Source Mask
    Source Port              any


    Destination Address
    Destination Mask
    Destination Port         any


               SAVE                    CANCEL

Use <Space> to select
```

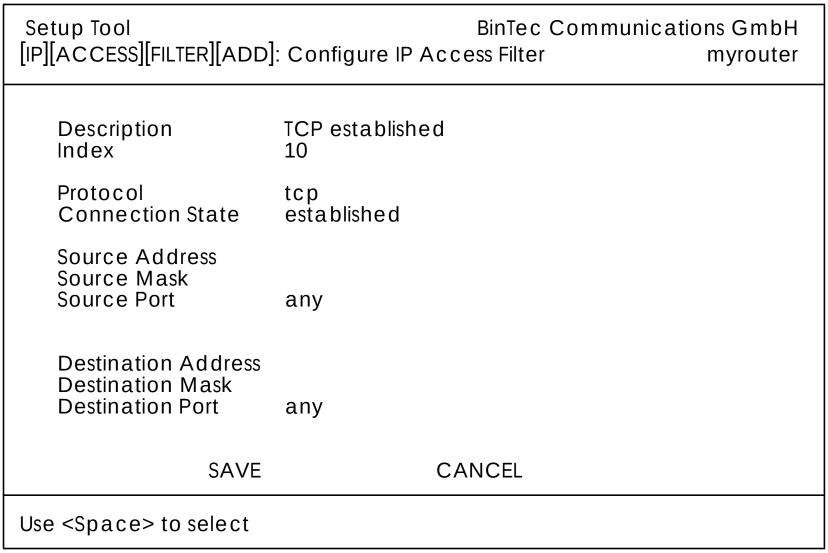
TCP Connection State

Filters can now be defined based on the state of an TCP Connection.
In the *ipFilterTable* there is the new variable *TcpConnState*, which can be assigned the following values:
dont_verify, established.
When this variable is set to *established*, this filter matches for TCP packets,which do not initiate a connection.
A typical application for this filter is to let packets pass through, which belong to connections that were initiated from inside, but discard all other TCP packets. This can be configured by the following rules:

1. rule:	ALLOW (TCP / established)
2. rule:	DENY (TCP / dont_verify)

The configuration in Setup Tool:

IP → ACCESS LISTS → FILTERS → ADD



```
 Setup Tool                                        BinTec Communications GmbH
[IP][ACCESS][FILTER][ADD]: Configure IP Access Filter                myrouter

     Description              TCP established
     Index                    10

     Protocol                 tcp
     Connection State         established

     Source Address
     Source Mask
     Source Port              any


     Destination Address
     Destination Mask
     Destination Port         any

                   SAVE                    CANCEL

Use <Space> to select
```

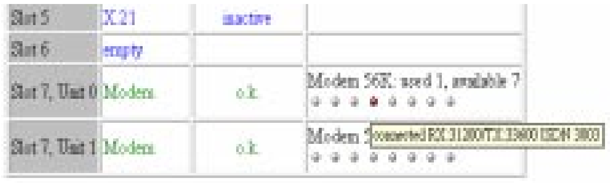
New Trace Command Feature

The trace command has been enhanced. It is now possible to decode HOLD and RETRIEVE messages in the D-Channel.

Status Display for Modems

The web based status page for your BRICK now additionally displays information on installed modems.

Under Hardware Interfaces you will find the modems with the respective slot they are installed in. The modem type and modem status are displayed. Each modem used at the moment is marked red and when you move the mouse pointer over the red channel symbol, the rate for receiving and transmitting data in bps is displayed. Additionally you are informed about the ISDN channel used.



Slot 5	X.21	inactive	
Slot 6	empty		
Slot 7, Unit 0	Modem	o.k.	Modem 56K: used 1, available 7 ○ ○ ○ ● ○ ○ ○ ○
Slot 7, Unit 1	Modem	o.k.	Modem 5 connected RX 31200 TX 33600 ISDN 3003 ○ ○ ○ ○ ○ ○ ○ ○

Wildcards for Called Party’s Address

Similar to wildcards for the calling party’s address in the *biboDialTable* for incoming calls, the variable *Number* now can also contain wildcards for outgoing calls. The wildcards for outoing and incoming calls are defined as follows:

Wildcard	Example	Outgoing Calls	Incoming Calls
*	1234*	is ignored, e.g 1234	matches zero or any string, e.g 1234 or 123467
?	1234?	is replaced by 0, e.g. 12340	matches any single digit, e.g. 12349, 12347

Wildcard	Example	Outgoing Calls	Incoming Calls
[a-b]	123[5-9]	first digit in the range, e.g. 1235	denotes the range of possible digits to match, e.g. 12345, 12346
[^a-b]	123[^0-5]	range of digits not allowed, first possible digit inserted, e.g. 1236	denotes the range of excluded digits to match, e.g. 12346, 12347
{ab}	{00}1234	inserted for outgoing calls, e.g. 001234	optional string to match, e.g. 001234, 1234

Link Quality Monitoring

By the help of Link Quality Monitoring (LQM defined in RFC 1989) it is possible to exchange information within a PPP connection to draw conclusions about the underlying connection quality.

This information is typically transmitted periodically to the partner as so-called Link Quality Reports (LQR). The interval (Reporting Period) is agreed upon during the LCP negotiation.

Link Quality Monitoring can be useful to examine e.g. modem connections. (With unreliable modem connections it can happen that because of CRC errors no more data can be transmitted.)

For detailed information on the new feature Link Quality Monitoring see the section Detailed Feature Descriptions on page 49.

Microsoft Point-to-Point Compression Protocol

The Microsoft Point-to-Point Compression Protocol (MPPC specified in RFC 2118) is based on a LZ based compression algorithm, which was optimized especially with regard to a large number of simultaneous connections and a MTU of 1500 Bytes (typical for PPP).

Because e.g. Windows NT Clients do not support the Stac LZS algorithm, it is possible to use MPPC to transmit compressed data also on these connections. Especially recent FM-

Stac modules support MPPC. There is no separate license required.

MPPC is activated by setting the variable *biboPPPCompression* in the *biboPPPTable* to *biboPPPCompression_mppc*.

Please note that MPPC can also be used in connection with MPPE, but this will result in disproportionately using the system to capacity for each B-Channel.

Bugfixes

Access Lists

- There was a bug in the access lists implementation concerning the following conditions:

  1.Filtering for source or destination ports.
  2.The action is deny
  3.The IP datagrams are fragmented.

  Under these conditions fragments are discarded, though the ports of the complete datagram do not correspond to the ports defined in the filters.

  This bug has been fixed.

TP0 Bridge

- Sometimes it happened that it was not possible to establish an initial TP0 Bridge connection (RFC 1086) between the TCP client and the BRICK.

  This bug has been fixed. In this context the TP0 Bridge syslog messages have been changed ,too.

X.31 in D-Channel

- When configuring TEI values for X.31 in D-Channel by the autoconfiguration for the BRICK's ISDN interface, it sometimes happend that wrong TEI values were generated in the *isdnDChanX31Table*.

This bug has been fixed.

- When CAPI applications were using X.31 in D-Channel for X.25 packet switching, the variable *AssignedTo* in the *isdnDChanX31Table* was not considered (Setup Tool: Advanced settings for the WAN interface).

  This problem has been fixed.

  The variable *AssignedTo* can be assigned the following values:

packet_switch	The TEI may be used only by the X.25 router.
capi	The TEI may only be used by a CAPI application and this TEI has to be configured also in the CAPI application.
capi_default	The TEI may be used only by a CAPI application and the BRICK overwrites the TEI value that is configured in the application.
delete	Sets the table entry to delete.

CAPI

- If a CAPI application used an X.25 protocol, it wasn't any longer informed about incoming X.25 calls after it has sent a CONNECT_B3_REQ message to establish an outgoing X.25 connection. In this case the CONNECT_B3_Ind message, the application has to receive got lost.

  This bug has been fixed.

Dynamic Shorthold

- When combining dynamic B-Channel Bundling with optimally making use of the charging intervals (by dynamic shorthold), there was the problem that, when reducing the bandwith, the current charging intervals were not taken into consideration. This problem has been fixed.
  To optimize charges now the bandwith is reduced by disconnecting a B-Channel only short before a new charging interval.

STAC Compression on Multilink PPP Interfaces

- According to RFC 1974 (*PPP STAC LZS Compression Protocol*) there are several check modes to keep the compressor and decompressor histories in synchronisation even in the absence of a reliable link to guarantee the sequential transmission of data.

  In Release 4.8.3 it still happened that in rare cases, when *biboPPPCompression* = stac (RFC 1974, check mode 3) was used on MultiLink PPP interfaces the history re-synchronisation process sometimes came to a state where decompression histories were out-of-sync and user data could no longer be transmitted over the line.

  This problem has been fixed in the current release.

Spaces in biboPPPLoginString

- There appeared problems in the login procedure configuration (especially for Compuserve users), when strings like passwords or login names were containing spaces. That was because spaces are used as internal flags to handle the login procedure.

  To handle this problem blank spaces in strings, which are part of the variable *LoginString* in the *biboPPPTable*, must be preceded by a backslash as shown in the following example for the string “pass word”:

```
inx LoginString(rw)
00 "-d1 \n e: CIS\n D: name/go:pppconnect\n wor -d1 pass\ word\n PPP"
```

Setting Administration Status to Down

When by “ifconfig down” or via the Setup Tool the variable *ifAdminStatus* was set to down for an active interface, the variables in the PPP accounting syslog message containing PPP connection information only had the value 0.

This problem has been solved. Now the syslog message contains the correct values.

Detailed Feature Descriptions

IPX RADIUS Extensions

The BRICK now supports dial-up IPX client connections via RADIUS. Support for IPX links via RADIUS has been tested using Merit's AAA RADIUS server 3.5.6. The examples shown below can be used with the Merit server, for use with other servers consult your local documentation.

Assuming a RADIUS server has been configured in Setup Tool's IP → RADIUS SERVER menu (or the *radiusServerTable* from the SNMP shell), and the RADIUS server can successfully authenticate the caller, dial-up links can be setup to support IPX networking using standard IPX or BinTec-specific RADIUS attributes mentioned below.

Standard Attributes

Framed-IPX-Network

This attribute defines a transfer network for the IPX link. Setting this attribute to "Framed-IPX-Network = 8" effectively sets the IPX network number for the transfer network to "0:0:0:8".

Normally, when this attribute is set to "ffffffe" the calling host is assigned a network number from an existing address pool, currently this feature is not supported on the BRICK. To disable a transfer network for the IPX WAN link you can set this attribute to "0" (no transfer network, unnumbered RIP).

RIP / SAP updates: If the "Framed-IPX-Network" attribute is used entries in the BRICK's *ripCircTable* and *sapCircTable* are configured using default settings for RIP / SAP updates (triggered+piggybacked).

BinTec-specific Attributes

BinTec-specific RADIUS attributes added in release 4.8.3 are defined below. For additional information, see also the section *Using BinTec-specific Attributes* on page 19.

BinTec-ipxCircTable	Creates or modifies ***ipxCircTable*** entries.

BinTec-ripCircTable	Creates or modifies ***ripCircTable*** entries
BinTec-sapCircTable	Creates ore modifies ***sapCircTable*** entries.

Example IPX over RADIUS /etc/raddb/client Entries

The definitions below could be used for a dial-up IPX link between two BRICKs. RIP / SAP updates will be performed via the default triggered+piggybacked mode.

```
kornburgxl
     Password = "access2roth", Framed-IPX-Network = 0
     Framed-Protocol = PPP, Idle-Timeout = 300,
rothxl
     Password = "access2kornburg", Framed-IPX-Network = 0
     Framed-Protocol = PPP, Idle-Timeout = 300,
```

If the router rothxl supports IPX WAN links but requires a transfer network (0:0:0:9) this definition could be used instead.

```
rothxl
     Password = "access2kornburg", Framed-IPX-Network = 9
     Framed-Protocol = PPP, Idle-Timeout = 300,
```

If the required IPX services are statically configured, RIP and SAP can be disabled for the WAN link using the BinTec-specific options shown below.

```
kornburgxl
     Password = "access2roth", Framed-IPX-Network = 0
     Framed-Protocol = PPP, Idle-Timeout = 300,
     BinTec-ripCircTable = "ripcircstate=off",
     BinTec-sapCircTable = "sapcircstate=off"
```

The definition below could be used for a dial-in Windows Client that does not support IPX WAN links. In this example, setting "Update = 0" disables periodic updates for active links.

```
winlaptop
     Password = "486pentium?"
     Framed-Protocol = PPP, Idle-Timeout = 300,
     Bintec-ipxCircTable = "netnumber=0:0:0:a ipxcirctype=ipxcpWS"
     BinTec-ripCircTable = "Update=0 AgeMultiplier=10000",
     BinTec-sapCircTable = "Update=0 AgeMultiplier=10000"
```

RIP/SAP Updates for RADIUS interfaces

Since RADIUS interfaces are only available as long as the respective client is connected please note the following effects this may have on links configured for active RIP and SAP updates.

- Access to services on the dial-in client's LAN (from hosts on the RADIUS client's LAN) may not be reliable.
- IPX clients cannot be informed of changes (routing or service advertisements) unless they are actually connected when the change occurs.

This may lead to a state where a server appears as being present on the remote network but is no longer available. The preferred solution, albeit time-consuming, to this problem is to statically configure the required routes and services on the clients and to disable RIP / SAP updates.

Using BinTec-specific Attributes

Each of the BinTec-specific RADIUS attributes corresponds to a MIB table. Supported BinTec-specific attributes can be used in your server's /etc/raddb/users file. The attribute definitions must also be added to your dictionary file (normally found in /etc/raddb). To modify a MIB table entry you must use the following syntax:

<BinTec-Option> = "*variable*1=*value*1 … *variable*n=*value*n"

An example authentication line from a RADIUS /etc/raddb/users file might look like this:

```
Service-Type = Framed,
BinTec-biboPPPTable = "DynShorthold=50 IpAddress=static",
BinTec-ipNatPresetTable = "Protocol=tcp extport=1050 intport=100"
```

Also, when using these attributes please note:

- The table entry's *ifIndex* is set automatically and can't be influenced.
- The entries are not case-sensitive.
- You must not use blank spaces before or after »=« signs inside the double quotes.
- Attributes support either **static** or **dynamic** mode.

Static mode modifies existing table entries while dynamic mode creates a new table entry. All variables you want to create (dynamic) must be defined in one line.

Frame Relay

Frame relay is a connection oriented technology that provides a fast packet-switching service for access to Wide Area Networks. It makes optimum use of available bandwidth using a complex statistical multiplexing algorithm. Due to the ommitance of some layer three network functions, Frame Relay is often thought of as a “streamlined version for X.25”.

Frame Relay is a flexible and cost-effective alternative to existing WAN technologies best suited for network installations exemplifying any of the following characteristics:

- Applications generate significant amounts of bursty-traffic
- Network traffic is delay-sensitive
- High network availability is a major priority
- Dispersed enterprise (locations separated by long distances)
- Integration with existing public and/or private packet switched networks is required.

The description of Frame Relay on the BRICK is broken down into the following sections.

1. An Overview of Frame Relay Technology (p. 20)
2. System tables for the Frame Relay subsystem (p. 38).
3. Setup Tool menus for configuring Frame Relay (p. 39).
4. Example Frame Relay configuration using Setup Tool (p. 49)
5. Example Frame Relay configuration using Setup Tool (p. 49)

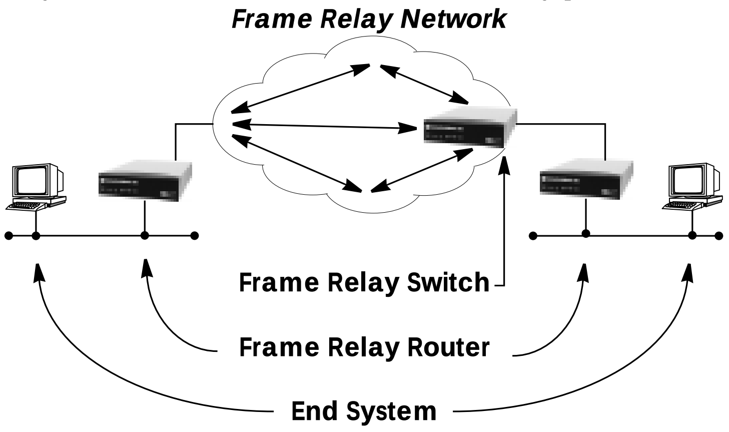
An Overview of Frame Relay Technology

As the name suggests, it works by breaking data streams into variable length frames and forwards these frames into the frame relay network via predetermined logical connections called **Permanent Virtual Circuits**, or PVCs.

Some of the key concepts of Frame Relay are listed below.

- Small, variable length frames are used to transport user data; this makes frame relay well suited for data applications (particularly those generating bursty-traffic) —video and voice transmissions are generally not appropriate.
- Improved overall performance (compared to X.25) —a result of limited error correction and acknowledgement routines.
- Users are guaranteed a minimum amount of bandwidth which is always available (the Committed Information Rate, or CIR).
- High network availability is achieved through statistically multiplexing virtual connections (data streams) onto logical connections, or Permanent Virtual Circuits (PVCs).
- Integrated bandwidth allocation (true bandwidth on demand) allows users to take up additional bandwidth, when available, at no extra charge —based on the user's Committed Burst Rate (CBR) and Excess Burst Rate (EBR).
- Congestion notification allows frame relay device to notify neighbouring devices (in either direction) of bandwidth bottlenecks to help main quality of services.

There are different types of equipment found in a typical Frame Relay Networks based on the various tasks they perform.



End Systems

End systems are typically end-user devices that take advantage (make use of) the underlying Frame Relay network. Depending on the application running on the end stations bandwidth requirements of end systems on the LAN can be different. Some applications generate large amounts of intermittent bursty traffic (typical of data applications, telnet, ftp, www) while others (like voice or video) require a constant bitrate.

Frame Relay Routers

Frame Relay Routers are used to connect point-to-multipoint networks (LANs) to a public (or private) Frame Relay network. Its the router's job to encapsulate data into Frame Relay frames for transport over the network link. A Frame Relay Router encapsulates LAN frames in frame relay frames and feeds those frames to a Frame Relay Switch for transmission across the network. A Frame Relay Router also receives frame relay frames from the network, strips the frame relay frame off each frame to product the original LAN frame, and passes the LAN frame on to the end device. A Frame Relay Router communicates directly with one or more Frame Relay Switches to negotiate the opening / closing of virtual circuits and to control network congestion.

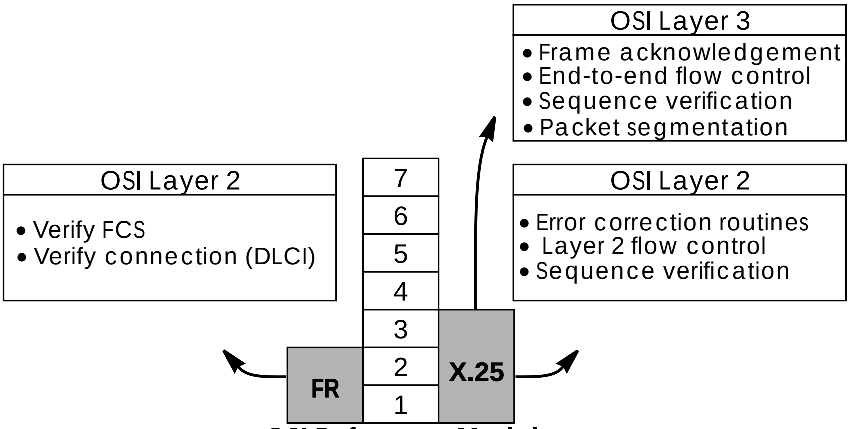
Frame Relay Switches

Switches are typically owned by public network providers but may be owned by private sites implementing private Frame Relay Networks. Aside from the FECN, BECN, and DE frame fields (used for congestion management) the content and final destination of individual frame is of no interest to the switch. Using a simple mapping scheme frames are passed from one interface (DLCI) to another.

Protocol Structure

Frame Relay Protocol Stack

Although similar in concept to X.25, frame relay operates at layer 2 of the OSI reference model. This is where the main differences between the two lie. Frame relay simply leaves out the extensive error detection / correction and end-to-end flow control found in X.25.



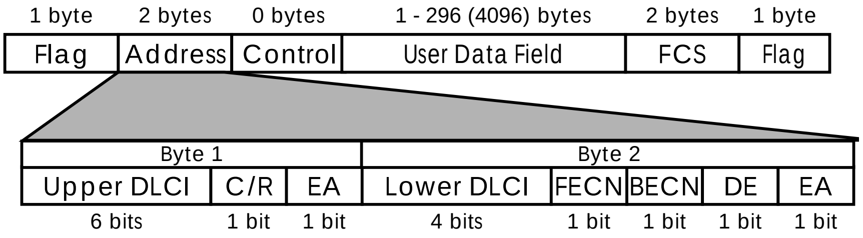
OSI Reference Model

This greatly simplifies the tasks a frame relay switch must perform.

Frame Relay Frame Format

As shown below frame relay is a streamlined protocol that uses HDLC framing. Virtual frame relay connections are routed based on the DLCI field of incoming frames.

Frame Relay Frame



Flag	HDLC Flag (bit sequence: 01111110)
FCS	Frame Checksum Sequence
DLCI	Data Link Connection Identifier
C/R	Command / Response Indicator
EA	Extended Address bit
FECN	Forward Explicit Congestion Notification
BECN	Backward Explicit Congestion Notification
DE	Discard Eligibility Indicator

Frame Relay Addressing

The basic (**unextended**) Frame Relay specification only supports locally significant addressing. These addresses are up to 2 bytes long. Using the EA fields **extended** addresses can be used which may be up to 4 bytes long.

When a frame is read the first EA bit that is set (i.e., it's value = 1) determines the address.

Congestion Notification

The FECN and BECN bits (see above) are used to notify neighbouring frame relay devices of possible congestion.

Virtual Circuits

In Frame Relay multiple connections are mapped to a single physical network connection.

Data Link Connection Identifier

The DLCI field is used to route virtual frame relay connections. A standard DLCI (2 byte address field) consists of 10 bits and is based on the frame's Upper and Lower DLCI fields. These 10 bits establish an upper limit of 1024, 2^{10}, possible simultaneous virtual channels that can be multiplexed on to a PVC.

A DLCI may specify a value between 0 and 1023; however not all values are valid. As shown below some values are reserved for network management or other features such as LAPD in the D-channel.

DLCI	Use (Q.922)	Use (LMI)
0	Signalling	Reserved
1- 15	Reserved	Reserved
16 - 511	Available (except when the D-channel is used)	Available
512 - 991	Available	Available
992 - 1007	Layer 2 management	Available
1008 - 1018	Reserved	Reserved
1019 - 1022	Reserved	Multicasting
1023	Consolidated Link Layer Management	Signalling

NOTE: A DLCI is only significant to the local station. Though it is used locally to identify both directions of a virtual circuit it has no meaning to the next station (or the destination) in the frame relay network.

Frame Relay Services

Frame relay access can be purchased in a variety of configurations depending of your site's needs. Characteristics of the service you will receive include:

1. The type of physical access you have to the frame relay network.
   You may have a 56 switched or 64k ISDN leased line.
2. The amount and type of bandwidth available; this will include your guaranteed and excess rates. See CIR, CBR, and EBR below.
3. The number of PVCs you are receiving.

Committed Information Rate

When purchasing frame relay services from your provider, you will be assigned a Committed Information Rate. This defines the minimum amount of bandwidth that your provider guarantees to be available to your site at all times.

Committed Burst Rate

You will also receive a Committed Burst Rate with your service package. This is an additional amount of bandwidth (in excess of your CIR) you may use when network resources are available. The CBR is free of charge, but be aware that all frames that are in excess of your CIR will be DE (Discard Eligible) flagged and may be discarded by intermediate switches if the network becomes congested.

Excess Burst Rate

As Excess Burst Rate is also available; it defines the maximum data rate the service provider's network will attempt to sustain. Also note that all EBR traffic is flagged Discard Eligible.

The Frame Relay Subsystem

Frame Relay on the BRICK consists of 5 SNMP system tables contained in the BRICK-XL's `fr` group. An overview of these tables is shown below. The full description of each SNMP object is contained on the following pages.

frGlobals	frDlcmiTable	frCircuitTable
frErrTable	frMprTable	

Overview: Frame Relay System Tables

- *frGlobals*
  Global settings for Frame Relay on the BRICK. Currently only contains the frTrapState object which is used to enabled / disable *frDLCIStatusChange* traps on the BRICK. (This trap indicates that the state of a particular Virtual Circuit has changed.)
- *frDlcmiTable*
  Contains parameters for each DLCM (Data Link Connection Management) interface for each instance of frame relay service on the BRICK.
- *frCircuitTable*
  Contains information for each Data Link Connection Identifiers and corresponding virtual circuits.
- *frErrTable*
  Used to store important status messages reported for interfaces configured with Link Management I (LMI).
- *frMprTable*
  Contains Multiprotocol Routing over Frame Relay interfaces (MPFR) on the BRICK. These interfaces are Virtual interfaces since they do not necessarily map to a single hardware interface. MPFR interfaces may be used by higher level protocols.

SNMP Objects: Frame Relay System Tables

frGlobals

This table is a static table consisting of one SNMP object as noted below.

- *frTrapState*

Default Value:	disabled
Description:	This variable indicates whether the system produces the *frDLCIStatusChange* trap.

frDlcmiTable

The Parameters for the Data Link Connection Management Interface for the frame relay service on this interface.

- *frDlcmiEntry*
  Default Value: none
  Description: The Parameters for a particular Data Link Connection Management Interface.

- *frDlcmiIfIndex*
  Default Value: none
  Description: The ifIndex value of the corresponding ifEntry.

- *frDlcmiState*
  Default Value: noLmiConfigured
  Description: This variable states which Data Link Connection Management scheme is active (and by implication, what DLCI it uses) on the Frame Relay interface. May be one of:
  noLmiConfigured, lmiRev1, ansiT1-617-D, q933a, or ansiT1-617-B.

- *frDlcmiAddress*
  Default Value: q922
  Description: This variable states which address format is in use on the Frame Relay interface.
  May be one of:
  q921, q922March90, q922November90,
  or q922.

- *frDlcmiAddressLen*
  Default Value: 2
  Description: This variable states which address length in octets. In the case of Q922 format, the length indicates the entire length of the address including the control portion.
  May be one of: two-octets, three-octets, or four-octets.

- *frDlcmiPollingInterval*
  Default Value: 10
  Description: This is the number of seconds between successive status enquiry messages.

- *frDlcmiFullEnquiryInterval*
  Default Value: 6
  Description: Number of status enquiry intervals that pass before issuance of a full status enquiry message.

- *frDlcmiErrorThreshold*
  Default Value: 3
  Description: This is the maximum number of unanswered Status Enquiries the equipment shall accept be fore declaring the interface down.

- *frDlcmiMonitoredEvents*
  Default Value: 4
  Description: This is the number of status polling intervals over which the error threshold is counted. For example, if within 'MonitoredEvents' number of events the station receives 'ErrorThreshold' number of errors, the interface is marked as down.

- *frDlcmiMaxSupportedVCs*
  Default Value: none
  Description: The maximum number of Virtual Circuits allowed for this interface. Usually dictated by the Frame Relay network.
  In response to a SET, if a value less than zero or higher than the agent's maximal capability is configured, the agent should respond bad Value

- *frDlcmiMulticast*

  Default Value: nonBroadcast

  Description: This indicates whether the Frame Relay inter face is using a multicast service. May be either broadcast or nonBroadcast.

- *frDlcmiDceIdleInterval*

  Default Value: 15

  Description: This is the number of seconds within a status enquiry messages should be received.

- *frDlcmiDceErrorThreshold*

  Default Value: 3

  Description: This is the maximum number of unsent Status Enquiries the equipment shall accept before declaring the interface down.

- *frDlcmiDceMonitoredEvents*

  Default Value: 4

  Description: This is the number of status polling intervals over which the error threshold is counted. For example, if within 'MonitoredEvents' number of events the station receives 'ErrorThreshold' number of errors, the interface is marked as down.

- *frDlcmiMode*

  Default Value: dte

  Description: This indicates the mode of the interface. dte(1) does status enquiry polling, dce(2) does status answering nni(3) is bidirectional use of both dte and dce.

- *frDlcmiDefaultRouteIfIndex*

  Default Value: 0

  Description: This is the default route interface index when a new PVC is established by means of line management.

frCircuitTable

A table containing information about specific Data Link Connection Identifiers and corresponding vitual circuits.

- *frCircuitIfIndex*
  Default Value: none
  Description: The ifIndex Value of the ifEntry this virtual circuit is layered onto.
- *frCircuitDlci*
  Default Value: none
  Description: The Data Link Connection Identifier for this virtual circuit.
- *frCircuitState*
  Default Value: active
  Description: Indicates whether the particular virtual circuit is operational. In the absence of a Data Link Connection Management Interface, virtual circuit entries (rows) may be created by setting virtual circuit state to ‘active’, or deleted by changing Circuit state to ‘invalid’. Whether or not the row actually disappears is left to the implementation, so this object may actually read as ‘invalid’ for some arbitrary length of time. It is also legal to set the state of a virtual circuit to ‘inactive’ to temporarily disable a given circuit. May be on of: invalid, delete, active, or inactive.
- *frCircuitReceivedFECNs*
  Default Value: none
  Description: Number of frames received from the network indicating forward congestion since the virtual circuit was created.

- *frCircuitReceivedBECNs*
  Default Value: none
  Description: Number of frames received from the network indicating backward congestion since the virtual circuit was created.

- *frCircuitSentFrames*
  Default Value: none
  Description: The number of frames sent from this virtual circuit since it was created.

- *frCircuitSentOctets*
  Default Value: none
  Description: The number of octets sent from this virtual circuit since it was created.

- *frCircuitReceivedFrames*
  Default Value: none
  Description: Number of frames received over this virtual circuit since it was created.

- *frCircuitReceivedOctets*
  Default Value: none
  Description: Number of octets received over this virtual circuit since it was created.

- *frCircuitCreationTime*
  Default Value: none
  Description: The value of sysUpTime when the virtual circuit was created, whether by the Data Link Connection Management Interface or by a SetRequest.

- *frCircuitLastTimeChange*
  Default Value: none
  Description: The value of sysUpTime when last there was a change in the virtual circuit state.

- *frCircuitCommittedBurst*

  Default Value: 0

  Description: This variable indicates the maximum amount of data, in bits, that the network agrees to transfer under normal conditions, during the measurement interval.

- *frCircuitExcessBurst*

  Default Value: none

  Description: This variable indicates the maximum amount of uncommitted data bits that the network will attempt to deliver over the measurement interval.

  By default, if not configured when creating the entry, the Excess Information Burst Size is set to the value of ifSpeed.

- *frCircuitThroughput*

  Default Value: 0

  Description: Throughput is the average number of ‘Frame Relay Information Field’ bits transferred per second across a user network i terface in one direction, measured over the measurement interval.

  If the configured committed burst rate and throughput are both non-zero, the measurement interval:

  T=CommittedBurst/Throughput

  If the configured committed burst rate and throughput are both zero, the measurement interval:

  T=CommittedBurst/Throughput.

  If the configured committed burst rate and throughput are both zero, the measurement interval:

  T=frCircuitExcessBurst/ifSpeed.

- *frCircuitRouteMode*
  Default Value: local
  Description: Type of routing entry. This field is intened for informational purpose only.
- *frCircuitRouteIfIndex*
  Default Value: 0
  Description: The destination IfIndex is a pointer to either a second frame relay interface (frame relay switching) or to a frame relay virtual interface (multiprotocol routing).
- *frCircuitRouteDlci*
  Default Value: 0
  Description: The destination DLCI. When the circuit is used for switching, all packets are sent using this DLCI. This field is void for multiprotocol routing.
- *frCircuitRouteMetric*
  Default Value: 0
  Description: The routing metric.

frErrTable

A table containing information about Errors on the Frame Relay interface.

- *frErrIfIndex*
  Default Value: none
  Description: The ifIndex Value of the corresponding ifEntry.

- *frErrType*
  Default Value: none
  Description: The type of error that was last seen on this interface and may be one of:
  unknownError, receiveShort,
  receiveLong, illegalDLCI,
  unknownDLCI, dlcmiProtoErr,
  dlcmiUnknownIE, dlcmiSequenceErr,
  dlcmiUnknownRpt, or noErrorSinceReset

- *frErrData*
  Default Value: none
  Description: An octet string containing as much of the error packet as possible. As a minimum, it must contain the Q.922 Address or as much as was delivered. It is desirable to include all information up to the PDU.

- *frErrTime*
  Default Value: none
  Description: The value of sysUpTime at which the error was detected.

frMprTable

This table contains an entry for each frame relay DCE virtual interface. This interface is called virtual because it does not necessarily reflect to a single hardware interface; it is possible to either narrow (many hardware interfaces looking as a single virtual) or widen (a single hardware interface with several VCs is mapped to several virtual intefaces) the interface.

Entries in this table are created manually and will result to the creation of a multiprotocol routing interface in the ifTable. This interface can than be used by higher level protocols like ip, ipx and bridging.

The object frMprIfIndex shall be set to 0 for creation of new entries. The BRICK will allocate the next free interface index and assign it to frMprIfIndex.

Rows of this table can be deleted by setting the object frMprEncapsulation to the value delete.

- *frMprIfIndex*

Default Value:	0
Description:	This object contains an interface index and assigns the row to an entry in ifTable. When creating new entries in the table, the value of this object shall be set to 0. The next free ifIndex value is than allocated by the BRICK and assigned to the object. At the same time a new interface is created in the IfTable.

- *frMprEncapsulation*

Default Value:	mpr
Description:	This object specifies the encapsulation method to be used. If this object is set to mpr, the encapsulation method described in Q933a is used for the protocols IP, IPX and transparent bridging. This encapsulation is also explained in RFC 1490.

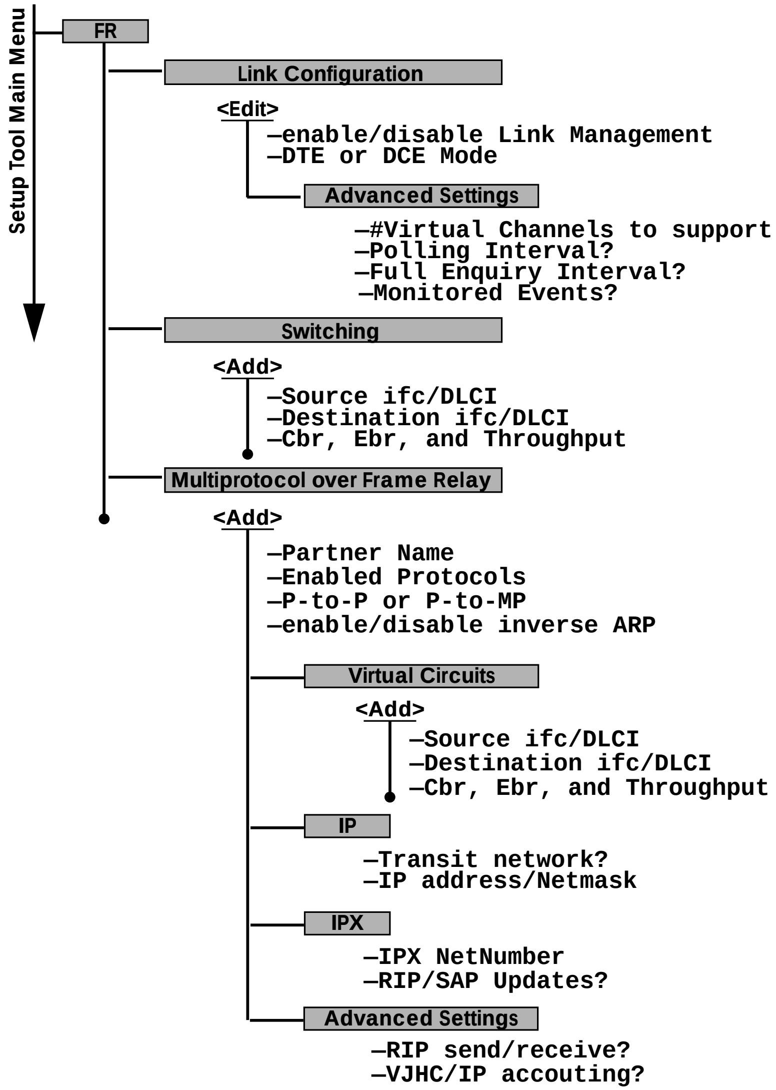
- *frMprMtu*
  Default Value: 1500
  Description: The Maximum transfer unit to be used with the interface (see ifMtu).
- *frMprIfcType*
  Default Value: point-to-point
  Description: This object specifies wether the interface is a multipoint(1) or a point-to-point(2) interface.
- *frMprInverseArp*
  Default Value: disabled
  Description: “This object specifies the use of Inverse Arp (as RFC 1490) to automatically detect upper layer network addresses.”

Frame Relay System Messages

biboAdmSyslogMessage	***~Level***
Attach link <*ifindex*> failed	debug
Attach link <*ifindex*>	debug
Bind link <*ifindex*> failed	debug
Link <*ifindex*> bound; starting LMI	debug
Be exceeded - packet discarded	debug
Want open ifc <*ifindex*>.	debug
Unknown ARP protocol <*proto*>	debug
No license	info
DLCI out of range: <*dlci*>	notice
No more than 256 interfaces allowed	error
Create: illegal index <*ifindex*>	error
Create: index <*ifindex*> already exists	error

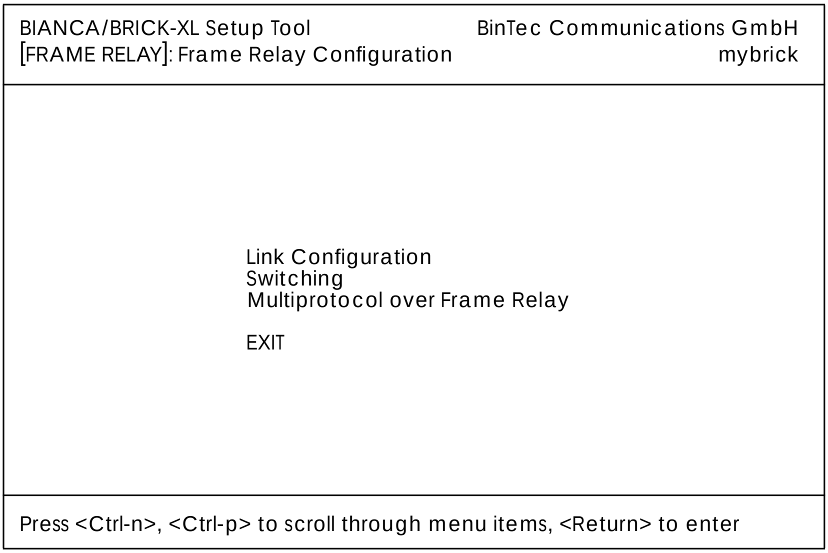
Frame Relay Setup Tool Menus

Several menus have to added to Setup Tool to allow for easy configuratin of Frame Relay on the BRICK. An overview of the menu structure is shown below. Individual submenus are described in detail on the following pages.



Setup Tool Menus

Frame Relay on the BRICK can be configured from Setup Tool using the three menus available here.



```
BIANCA/BRICK-XL Setup Tool                         BinTec Communications GmbH
[FRAME RELAY]: Frame Relay Configuration                              mybrick



                     Link Configuration
                     Switching
                     Multiprotocol over Frame Relay

                     EXIT



Press <Ctrl-n>, <Ctrl-p> to scroll through menu items, <Return> to enter
```

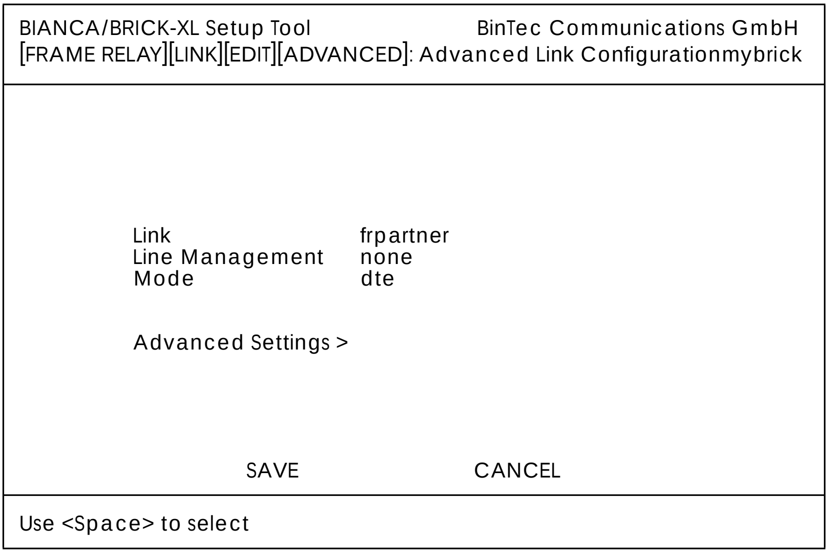
LINK CONFIGURATION contains the settings relative to the layer 2 of Frame Relay interface.

SWITCHING lists settings for each Frame Relay Virtual Circuit.

MULTIPROTOCOL OVER FRAME RELAY lists all existing MPFR interfaces configured on the BRICK.

FR → **LINK CONFIGURATION** →

This menu lists the available links that may be configured as the transport layer of a Frame Relay interface. Use the menu shown below (First select the link and hit enter) to edit link's settings.



```
BIANCA/BRICK-XL Setup Tool                        BinTec Communications GmbH
[FRAME RELAY][LINK][EDIT][ADVANCED]: Advanced Link Configurationmybrick



          Link                   frpartner
          Line Management        none
          Mode                   dte


          Advanced Settings >



                   SAVE                     CANCEL

Use <Space> to select
```

Link = Shows the link that is currently being edited.

Line Management = Determines whether or note link management is being performed on this link. Currently, the method described in Q.933 is supported.

Mode = Defines the mode (DTE or DCE) the BRICK operates at for this connection. Note that one side of the link must operate as DTE and one as DCE.

Select **SAVE** to accept the settings and return to the previous menu.

Select **CANCEL** to discard all changes made since the last SAVE and return to the previous menu.

FR → **LINK CONFIGURATION** → **ADVANCED SETTINGS** →

This menu defines several advanced settings relating to line management for Frame Relay interfaces on the BRICK . Some options only apply to BRICK soperating in DTE or DCE mode.

```
BIANCA/BRICK-XL Setup Tool                    BinTec Communications GmbH
[FRAME RELAY][LINK][EDIT][ADVANCED]: Advanced Link Configurationmybrick




      Supported Vrtual Channels        250

      Polling Interval                 10
      Full Enquiry Interval            6
      Idle Timer                       15
      Error Threshold                  3
      Monitored Events                 4




                     OK                     CANCEL

Enter integer range 1 ..250
```

Supported Virtual Channels =

Polling Interval = (only DTE with line mgmt)

Full Enquiry Interval = (only DTE with line mgmt)

Idle Timer = (only fpr DCE with line mgmt)

Error Threshold = (only with line mgmt)

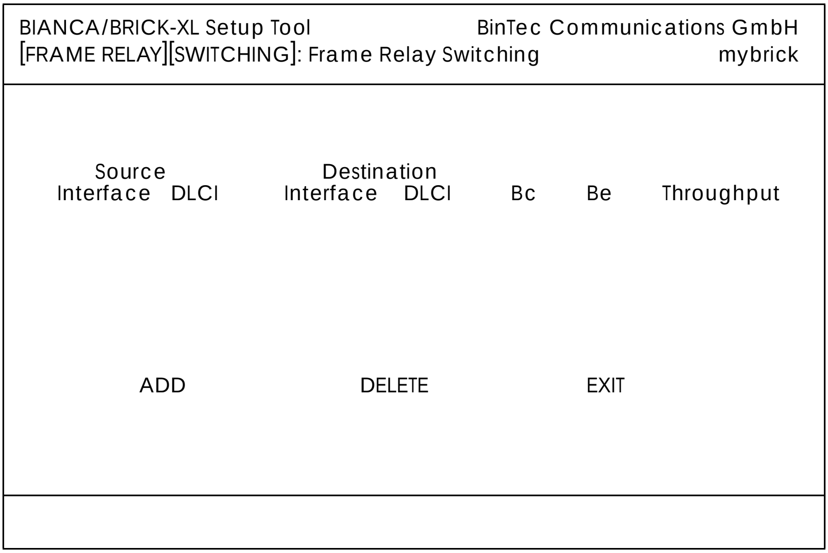
Monitored Events = (only with line mgmt)

Select **OK** to accept the settings and return to the previous menu.

Select **CANCEL** to discard all changes made since the last SAVE and return to the previous menu.

FR → **SWITCHING** →

This menu displays the current list of Frame Relay routes. Frame Relay routes can be added, removed, or changed here.



```
BIANCA/BRICK-XL Setup Tool                    BinTec Communications GmbH
[FRAME RELAY][SWITCHING]: Frame Relay Switching                  mybrick



        Source                  Destination
   Interface  DLCI         Interface  DLCI       Bc      Be     Throughput







          ADD                    DELETE                EXIT
```

Select **ADD** to create a new Frame Relay route.

Select **DELETE** to remove a Frame Relay route entry that has been tagged (using the spacebar) for deletion.

Select **EXIT** to accept the list of Frame Relay routes and return to the previous menu.

To edit a Frame Relay route, highlight the entry and then enter <Return>. When adding or changing an entry the following information must be provided.

Source Interface = Using the spacebar scroll through the list of Frame Relay interfaces.

Source DLCI =

Destination Interface =

Destination DLCI =

Committed Burst Rate =

Excess Burst Rate =

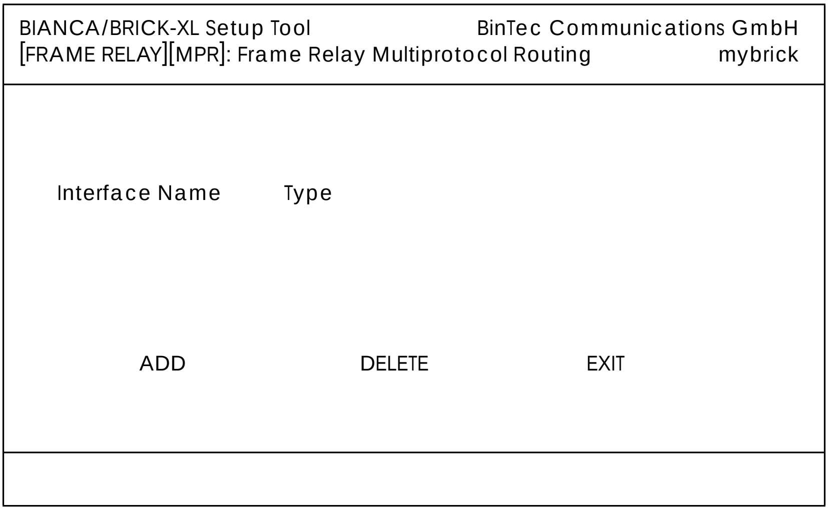
Throughput =

Select **OK** to accept the settings and return to the previous menu.

Select **CANCEL** to discard all changes made since the last SAVE and return to the previous menu.

FR → MULTIPROTOCOL OVER FRAME RELAY

This menu lists Multiporotocol Routing over Frame Relay interfaces on the BRICK. MPFR interfaces can be added, removed, or changed here. .



```
BIANCA/BRICK-XL Setup Tool                      BinTec Communications GmbH
[FRAME RELAY][MPR]: Frame Relay Multiprotocol Routing               mybrick


     Interface Name        Type




          ADD                  DELETE                  EXIT
```

Interface Name = Identifies the interface name (taken from the *ifDescr* object from the *ifTable*).

Type = Specifies whether the interface is a point-to-point, or point-to-multipoint interface.

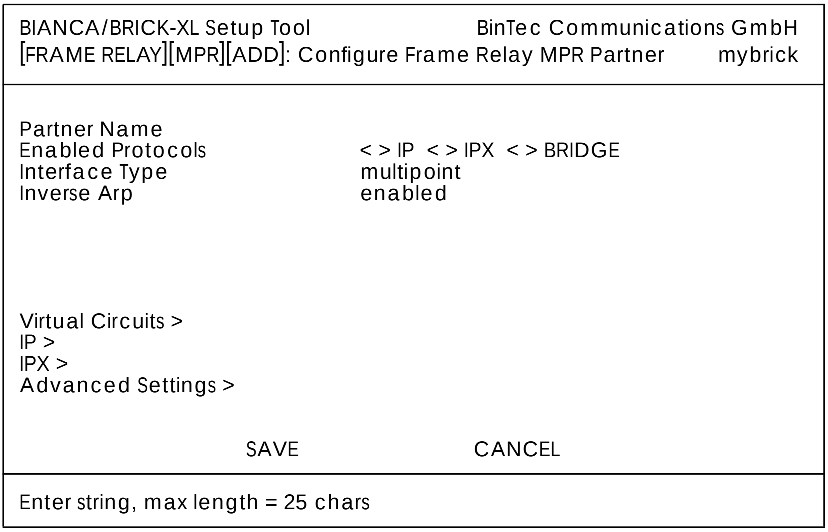
Select **ADD** to create a new MPFR interface. (See the EDIT / ADD menu on the following page.)

Select **DELETE** to remove a MPFR interface that has been tagged (using the spacebar) for deletion.

Select **EXIT** to accept the interface list and return to the previous menu.

FR → MULTIPROTOCOL OVER FRAME RELAY → ADD

This menu is used to create (or change) MPFR interfaces on the BRICK .



```
BIANCA/BRICK-XL Setup Tool                    BinTec Communications GmbH
[FRAME RELAY][MPR][ADD]: Configure Frame Relay MPR Partner       mybrick

Partner Name
Enabled Protocols               < > IP  < > IPX  < > BRIDGE
Interface Type                  multipoint
Inverse Arp                     enabled




Virtual Circuits >
IP >
IPX >
Advanced Settings >

                    SAVE                    CANCEL

Enter string, max length = 25 chars
```

Partner Name = A unique name to identify this FR partner.

Enabled Protocols = Identifies the protocols that will be supported by this link. Currently, only IP traffic is supported.

Interface Type = Determines the interface type as being either “multipoint” or “point to point”.

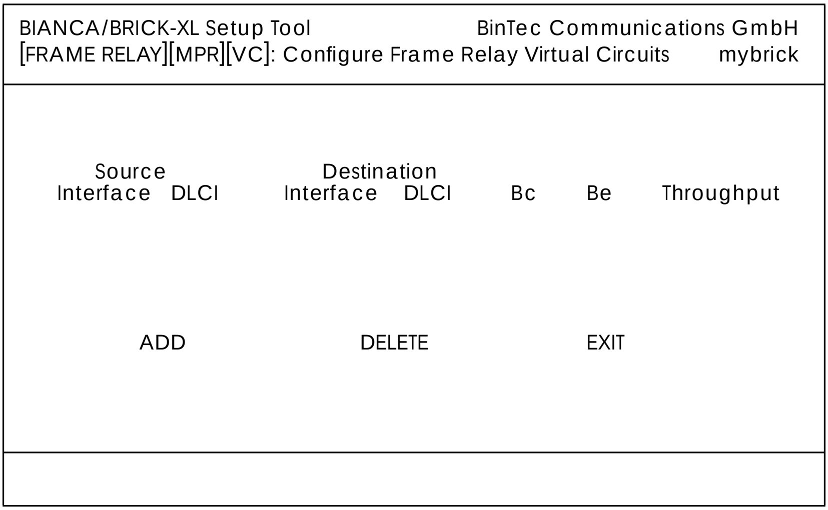
Inverse Arp = Enables / disables inverse ARP over this interface.

Select SAVE to accept the settings and return to the previous menu.

Select CANCEL to discard all changes made since the last SAVE and return to the previous menu.

FR → **MULTIPROTOCOL OVER FRAME RELAY** → **VIRTUAL CIRCUITS** →

This menu lists the Virtual Circuits configured for this Frame Relay interface. .



```
BIANCA/BRICK-XL Setup Tool                         BinTec Communications GmbH
[FRAME RELAY][MPR][VC]: Configure Frame Relay Virtual Circuits       mybrick



       Source                  Destination
   Interface   DLCI        Interface    DLCI      Bc      Be      Throughput





          ADD                    DELETE                 EXIT
```

Source Interface = Using the spacebar scroll through the list of Frame Relay interfaces.

Source DLCI =

Committed Burst Rate =

Excess Burst Rate =

Throughput =

Select **ADD** to create a new Virtual Circuit for this FR interface.

Select **DELETE** to remove an existing Virtual Circuit that has been tagged (using the spacebar) for deletion.

Select **EXIT** to accept the list and return to the previous menu.

FR → MULTIPROTOCOL OVER FR → ADD → IP →

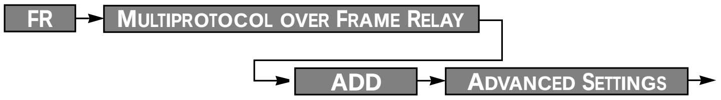
This is where you configure the IP settings for this remote MPX25 partner and is only available if the IP protocol has been enabled.

> **Note:** The settings used in this menu are the same as those used in the **WAN PARTNER → ADD → IP →** menu described in the *User's Guide* but only apply to this MPFR partner.

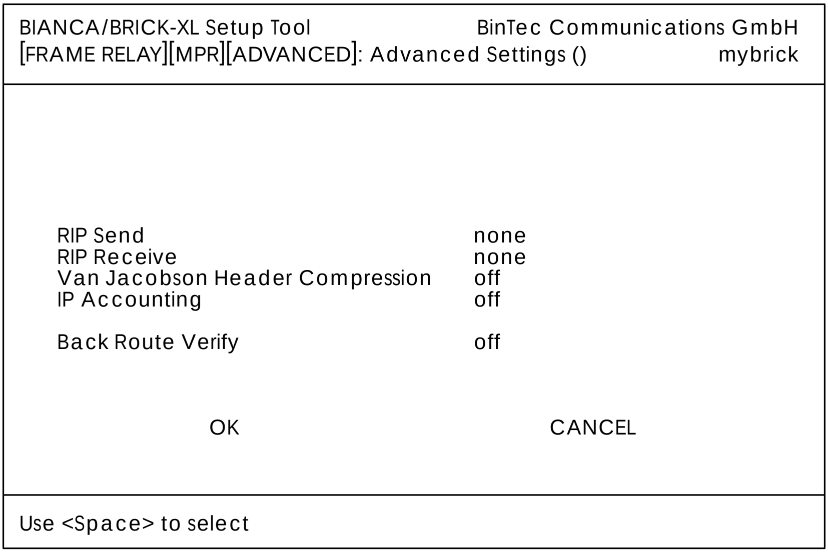
FR → MULTIPROTOCOL OVER FR → ADD → IPX →

This is where you configure the IPX settings for the remote MPFR partner. This menu is only available if IPX has been enabled.

> **Note:** The settings used in this menu are the same as those used in the **WAN PARTNER → ADD → IPX →** menu described in the *User's Guide* but only apply to this MPFR partner.



This menu is to used to configure several advanced features (a subset of the same features found in the WAN Partners Advanced Settings menu). .



```
BIANCA/BRICK-XL Setup Tool                          BinTec Communications GmbH
[FRAME RELAY][MPR][ADVANCED]: Advanced Settings ()                    mybrick




   RIP Send                                    none
   RIP Receive                                 none
   Van Jacobson Header Compression             off
   IP Accounting                               off

   Back Route Verify                           off



              OK                                     CANCEL



Use <Space> to select
```

RIP Send = This option allows you to control the types (if any) of RIP packets to transmit the frame relay interface.

RIP Receive = This option allows you to control the types (if any) of RIP packets to accept via the frame relay interface.

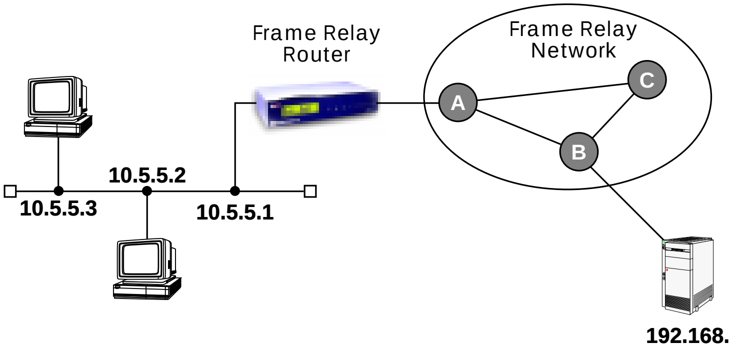
Van Jacobson Header Compression = Disable or enable IP header compression for packets send via this interface.

Back Route Verify = Optionally verify the sender’s IP address.

Example Configuration using Setup Tool

Frame Relay over Dialup Lines

The BRICK-XL shown below could be configured as a Frame Relay router as follows.



Requirements: Frame Relay requires a separate license to be installed. After installing your license verify Frame Relay subsystem is active in the biboLicenseTable.

1. Configure a WAN Partner
2. Configure the FR Link
3. Configure the MPR Interface

Link Quality Monitoring

Because Link Quality Monitoring as described under Features is specified within LCP negotiation, i.e. before the authentication of the partner, for the configuration of incoming calls a distinction must be made between inband and outband indentification.

In case of outband identification (CLID / outband RADIUS) and for outgoing calls the LQM is activated by setting the variable *biboPPPLQMonitoring* in the *biboPPPTable* to on. When a RADIUS server is used the variable is set by the help of the BinTec dictionary.

For incoming calls identified inband (identification by the internal *biboPPPTable* or via RADIUS server) the variable *biboPPPProfileLQMonitoring* in the *biboPPPProfileTable* must be set to on.

After a successful LCP negotiation for every link of a temporary connection additionally to the entry in the *biboPPPLinkTable* a correlating entry in the *biboPPPLQMTable* is generated. Both entries can be uniquely assigned to each other by the *IFIndex* respectively the *CallReference* value.

The *biboPPPLQMTable* is a new table and is described in detail in the following.
biboPPPLQMTable:

inx	IfIndex(*ro	CallReference(ro)	ReportingPeriod(ro)
	OutLQRs(ro)	OutPackets(ro)	OutOctets(ro)
	InLQRs(ro)	InPackets(ro)	InOctets(ro)
	InDiscards(ro)	InErrors(ro	PeerOutLQRs(ro)
	PeerOutPackets(ro)	PeerOutOctets(ro)	PeerInLQRs(ro)
	PeerInPackets(ro)	PeerInOctets(ro	PeerInDiscards(ro)
	PeerInErrors(ro	LossedOutLQRs(ro)	LossedOutPackets(ro)
	LossedOutOctets(ro)	LossedPeerOutLQRs(ro)	LossedPeerOutPkts(ro)
	LossedPeerOutOcts(ro)		

The *biboPPPLQMTable* contains statistical information for each current PPP link on the system. Only the system can add or delete entries to this table.

Entries are created by the system each time a new PPP link was established and LQM was negotiated successfully.

Entries are removed by the system, when the corresponding PPP link is disconnected.
For detailed information on the meaning of the single variables see the MIB Reference on the BinTec Website at http://www.bintec.de

What Was New in Release 4.8.3

Release 4.8 Revision 3:

Released: 24.04.98

Features: | Bugfixes: | Detailed Description:

Features

New CM-2XBRI Communications Module

Beginning with Revision 3 the BRICK-XL now supports BinTec's new CM-2XBRI Double S_0 with FAX/modem module. Highlights of the new CM-2XBRI are as follows.

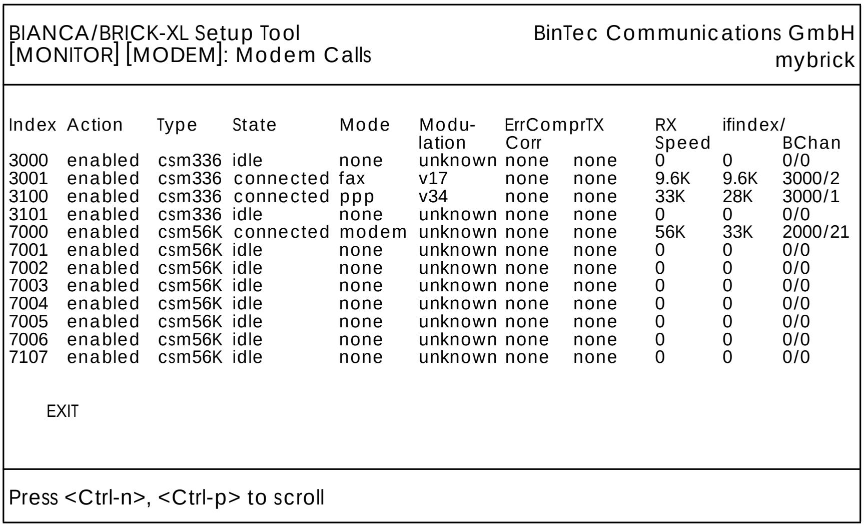
- Separate S_0 interfaces (Euro ISDN, 1TR6, NI1/2).
- Expandible Modem Pool: Up to four modems can be installed via modem daughtercards. Each daughtercard includes two V.34 data (33.6)/V.17 FAX modems.
- Resource Allocation: The CM-2XBRI's modems are not limited to a specific B-channel but are dynamically allocated to incoming analog calls received via any of this CM-2BRI's S_0 interfaces.
- Upgradable: The internal modem code for operating the CM-2XBRI's modems is integrated into flash memory located onboard. .

For information regarding installing/configuring the new CM-2XBRI module refer to the most recent version (1.3) of the *BRICK-XL User's Guide*. This document has been updated to reflect System Software Release 4.8 and is available via the URL: http://www.bintec.de/download/brick/doku/70015.pdf.

Monitoring Modem Connections

A new *mdmTable* has been added to the SNMP shell’s Modem Group. The *mdmTable* contains one row entry for each modem detected on the BRICK (via CM-2XBRI or FM-8MOD modules). Each entry includes information regarding the current state of a specific modem. Status information details the detected modem type, whether a connection is established, and if so, the associated ISDN B-channel, the transmit / receive rates, as well as the negotiated compression and modulation modes.

Setup Tool has also been enhanced in Revision 4.8.3 and displays information retrieved from the new *mdmTable* in the MONITORING AND DEBUGGING → MODEM → menu as shown below.



```
BIANCA/BRICK-XL Setup Tool                         BinTec Communications GmbH
[MONITOR] [MODEM]: Modem Calls                                         mybrick

Index Action   Type   State     Mode  Modu-   ErrComprTX    RX     ifindex/
                                      lation  Corr          Speed        BChan
3000  enabled  csm336 idle      none  unknown none   none   0      0     0/0
3001  enabled  csm336 connected fax   v17     none   none   9.6K   9.6K  3000/2
3100  enabled  csm336 connected ppp   v34     none   none   33K    28K   3000/1
3101  enabled  csm336 idle      none  unknown none   none   0      0     0/0
7000  enabled  csm56K connected modem unknown none   none   56K    33K   2000/21
7001  enabled  csm56K idle      none  unknown none   none   0      0     0/0
7002  enabled  csm56K idle      none  unknown none   none   0      0     0/0
7003  enabled  csm56K idle      none  unknown none   none   0      0     0/0
7004  enabled  csm56K idle      none  unknown none   none   0      0     0/0
7005  enabled  csm56K idle      none  unknown none   none   0      0     0/0
7006  enabled  csm56K idle      none  unknown none   none   0      0     0/0
7107  enabled  csm56K idle      none  unknown none   none   0      0     0/0

    EXIT

Press <Ctrl-n>, <Ctrl-p> to scroll
```

For detailed information regarding the new *mdmTable* or Setup Tool’s [Monitoring and Debugging][Modem] menu please refer to the *BIANCA/BRICK MIB Reference* and Chapter 4 of the *BRICK-XL User’s Guide* respectively (both are available via BinTec’s web site).

Changes

PPP

- **Reporting of Dynamically Assigned IP Addresses**:
  In previous releases syslog messages were generated when a remote client was assigned an IP address from the BRICK's local IP address pool (*biboPPPIpAssignTable*).

  If a host-route is configured on the BRICK for the calling client, the BRICK always retrieves the address from the host route before checking the address pool. If an address was assigned in this manner a syslog message was not generated. Beginning in release 4.8.3 syslog messages are generated for all IP address assignments regardless of the method used.

Bugfixes

FAX

- In previous releases FAX connections (via FM-8MOD) were sometimes inadvertently lost during page breaks while sending or receiving longer (multi-page) facsimiles. This problem has been corrected in revision 3.

IP

- In previous releases telnet sessions from the BRICK to hosts supporting the "Telnet Data Encryption Option" (typically supported on BSD UNIX systems) was not possible. This problem has been fixed in Revision 3.
- In rare cases it wasn't possible to delete entries from the BRICK's *ipNatOutTable*. This problem has been fixed.

IPX

- IPX WAN links that were configured using the setting "piggyback (only if link active)" in the **Send RIP/SAP Updates** field in Setup Tool's [WAN Partners][EDIT][IPX] menu sometimes unexpectedly caused a system reboot when many changes occurred on the network and the

WAN link had not been active for a long time. This problem has been fixed.

PPP

- **PPP Accounting**:
  In previous releases setting the *IfAdminStatus* object to "down" for an interface that was in the up state sometimes resulted in syslog accounting messages containing incorrect data.

  Before:

  ```
  "dialup1: outgoing link closed, duration 0 sec,
          0 bytes received, 0 bytes sent, 0 charging units"
  ```

  After:

  ```
  "dialup1: outgoing link closed, duration 45 sec,
          2365 bytes received, 4347 bytes sent, 3 charging units
  ```

- **STAC Compression on MultiLink PPP Interfaces**:
  According to RFC 1974 (*PPP STAC LZS Compression Protocol*) there are several check modes to keep the compressor and decompressor histories in synchronisation even in the absence of a reliable link to guarantee the sequential transmission of data.

  In rare cases, when *biboPPPCompression* = stac (RFC 1974, check mode 3) was used on MultiLink PPP interfaces the history re-synchronisation process sometimes came to a state where decompression histories were out-of-sync and user data could no longer be transmitted over the line.

  Although the frequency of this problem has been greatly reduced in release 4.8.3 we currently recommend using MS-STAC compression (*biboPPPCompression*=ms_stac) when the remote partner supports this method. If the partner interface does not support MS-STAC, please note that this problem only occurs when MultiLink PPP is configured; and even then only in rare cases.

SNMP

- A problem involving the *biboAdmLoginTable* has been fixed in revision 3. If a failed login attempt occurred on the BRICK for a user that was defined in this table (either an incorrect password or no password at all was entered) the BRICK automatically prompted with a new **`login:`** string. If the admin, read, or write user was then entered followed by the appropriate password, the BRICK incorrectly started the command (*biboAdmLoginCommand*) configured for the user from the failed login instead of the SNMP shell session. This problem has been fixed.

Detailed Feature Descriptions

For information regarding the new features included in System Software Release 4.8 Revision 3 please refer to the most recent versions of user documentation for the BIANCA / BRICK-XL.

User documentation is available (in Adobe's PDF format) via BinTec's Internet web site at:

http://www.bintec.de/ftp/brick_xl.html.