http://www.bintec.de

RELEASE NOTE BIANCA/BRICK-XM

December 5, 1997

New System Software:
Release 4.6 Revision 4

This document describes the new features, enhancements, bugfixes, and changes to the BIANCA/BRICK-XM System Software since Release 4.5 Revision 5.

Upgrading System Software

1. Retrieve the current system software image from BinTec's HTTP server at http://www.bintec.de.
2. With this image you can upgrade the BIANCA/BRICK-XM 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-XM User's Guide* under *Firmware Upgrades*.
3. Once you've installed Release 4.6 Revision 4 you may want to retrieve the latest documentation (in Adobe's PDF format) which is also available from BinTec's FTP server 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 Revision 4

Release 4.6 Revision 4: Released: 05.12.97

Features

Multiple RADIUS Servers

With Revision 4 you can now configure more than one RADIUS server. For this purpose the new ***radiusServerTable*** was added to the administration group. For a detailed description of the ***radiusServerTable*** please refer to section MIB changes on page .

You can, of course, load a configuration file from an older release in a BRICK with Rel. 4.6 Rev. 4. A *biboAdmRadiusServer* entry from this file will be transformed to a corresponding ***radiusServerTable*** entry on bootup.
On the other hand, if you use a configuration file created with Rel. 4.6 Rev. 4 on a BRICK with an earlier system software release you will loose your RADIUS Server configuration.

The Setup Tool now contains the new [IP][Radius Server] menu, which lists all the RADIUS Servers currently configured. You can add, edit, or delete list entries in the usual fashion.

For each Radius Server you can configure the following parameters:

Protocol = Use this RADIUS Server for authentication purposes (**auth**) or for accounting ISDN connections (**acct**).
When you configure a RADIUS Server for accounting, the BRICK transmits Start and Stop Radius packets for each ISDN connection to this server.
Default value: auth

IP Address = IP Address of the RADIUS Server.

Password = Shared secret between RADIUS Server and BRICK.

```
BIANCA/BRICK-XM Setup Tool                    BinTec Communications GmbH
[IP][RADIUS][EDIT]: Configure Radius Server                      mybrick

 Protocol                auth

  IP Address             44.55.66.77
  Password               blubb

  Priority               0
  Policy                 authoritative

  Port                   1812
  Timeout                1000
  Retries                1
  State                  active

             SAVE                           CANCEL

Use <Space> to select
```

Priority = 0 … 7. When there are several RADIUS Server entries, the server with the lowest priority entry is used first. If there is no reply from this server, the server with the next lowest priority entry is used, and so forth, i.e. servers with *Priority*=**0** have the highest priority.
Default value: 0

Policy = can be set to **authoritative** or **non-authoritative**. If set to authoritative, a negative answer to a request will be accepted. This is not necessarily true when set to **non-authoritative**, where the next radius server will be asked until there is finally an **authoritative** server configured.
Default value: authoritative

Port = TCP port to use for RADIUS data. According to RFC 2138 the default ports are 1812 for authentication (was 1645 in older RFCs) and 1813 for accounting (1646 in older RFCs).
Default value: 1812

Timeout = 50 … 50000, number of milliseconds to wait for an answer to a request.
Default value: 1000 (1 second)

Retries = number of retries if a request is not answered. If after *Retries* attempts still no answer was received, the server *State* is set to **inactive**. The BRICK then tries to contact the

Server every 20 seconds, and once the Server replies, the *State* is changed to **active** again.
Default value: 1

State = the state of the RADIUS Server. In normal operation mode this is either **active** (server answers requests) or **inactive** (server does not answer; see *Retries* above). You can also set State=**disabled**, to temporarily disable requests to a certain RADIUS Server.
Default value: active

MIB Changes

New *radiusServerTable*

The variables and their possible values of the ***radiusServerTable*** are listed in the following table. For a description of these values please refer to the corresponding explanations of the Setup Tool menu above.

Variable	**Possible Values**
Protocol (*rw)	authentication, accounting
Address (rw)	IP Address
Port (rw)	0...65535
Secret (rw)	RADIUS Password
Priority (rw)	0...7
Timeout (rw)	50...50000
Retries (rw)	0...10
State (-rw)	active, inactive, disabled, delete
Policy (rw)	authoritative, non_authoritative

Microsoft Compatible CHAP (MS-CHAP)

Microsoft's proprietary CHAP implementation (Microsoft PPP CHAP Extensions, Revision 1.3) is now supported. PPP partners that use MS-CHAP authentication should now be configured by setting ***biboPPPAuthentication*** to **`ms-chap`**, or by selecting MS-CHAP in the "PPP Authentication Protocol" field in Setup Tools's [WAN Partners] menu.

Another encapsulation method (**`all`**) has also been added to ***biboPPPAuthentication*** to allow MS-CHAP (as well as CHAP and PAP) authentication to be attempted. As of Revision 4 the following authentication options my be configured.

biboPPPAuthentication	Attempted Authentication Methods[a]
`none`	No inband authentication is performed.
`pap`	Authentication ONLY via PAP.
`chap`	Authentication ONLY via CHAP.
`both`	Authentication via CHAP OR PAP.
`radius`	Authentication via biboAdmRadiusServer using CHAP OR PAP.
`ms-chap`	Authentication ONLY via MS-CHAP.
`all`	Authentication via CHAP, MS-CHAP, OR PAP.

a. In the case of multiple protocols authentication is attempted in the order mentioned.

Note: Most older RADIUS implementations do not support MS-CHAP authentication.

Configurable PPP Defaults

With Revision 4 the default settings used for dial-in PPP connections are configurable using the new ***biboPPPProfileTable***. These settings only apply to incoming connections from callers that can't be verified via CLID (Calling Party's Number).

The new ***biboPPPProfileTable*** consists of three fields.

`Name``(*ro)`	–Read-only; the profile name.
`AuthProtocol``(rw)`	–Defines the default protocol to use for PPP authentication.
`AuthRadius``(rw)`	–Controls the use of a RADIUS server when default PPP settings are applied.

See Default PPP Handling for detailed information on how the default PPP settings are determined for incoming PPP connections.

Bugfixes

PMX

- When using V.110 on a PMX board under certain conditions wrong data bytes were transmitted. This bug has been fixed.

PPP

- The authentication entry of a WAN partner was mistakenly set to **radius** when a second call came in and multilink PPP was not activated.
  This bug has been fixed.

- Partner specific local PPP identification (***biboPPPLocalIdent***) can now be used in connection with any authentication protocol (***biboPPPAuthentication*** = CHAP, MS-CHAP, or PAP).

  In previous releases there was a limitation involving callback negotiation and inband CHAP authentication.

CAPI

- Under certain conditions the BRICK did not create a »Sending Complete« indication. This bug has been fixed.

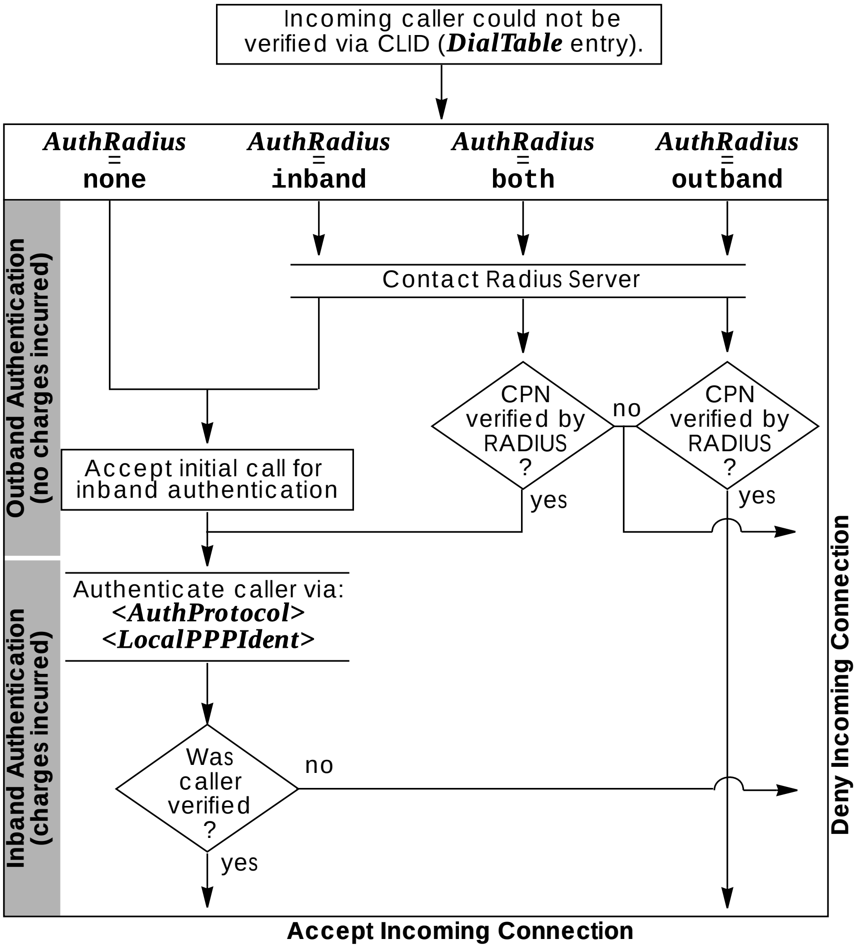
- In previous releases, CAPI applications that were initially notified of an incoming call (CONNECT_IND) but did not actually receive the call were incorrectly sent a “Normal call clearing” (DISCONNECT_IND) message. This has been corrected. Applications now receive an “Another application got the call.” message (CAPI 2.0) or an “Abort D-Channel layer 1” message (CAPI 1.1).

Detailed Feature Descriptions

Default PPP Handling

The diagram below shows how the BRICK's default PPP settings are determined using the settings found in the ***biboPPPProfileTable***.

Note: These settings ONLY affect the default handling of INCOMING connections that couldn't be identified by the Calling Party's Number.


Incoming caller could not be verified via CLID (*DialTable* entry).
AuthRadius = none
AuthRadius = inband
AuthRadius = both
AuthRadius = outband
Outband Authentication (no charges incurred)
Inband Authentication (charges incurred)
Contact Radius Server
CPN verified by RADIUS ?
no
CPN verified by RADIUS ?
yes
yes
Accept initial call for inband authentication
Authenticate caller via: <*AuthProtocol*> <*LocalPPPIdent*>
Was caller verified ?
no
yes
Deny Incoming Connection
Accept Incoming Connection

Release 4.6 Revision 2:

Released: 10.11.97

Features

Auto-Logout Timer

A new Auto-Logout timer has been implemented on the BRICK. This feature means that each login session started on the BRICK now automatically times out after a pre-configured timer expires (by default 900 seconds).

This feature has been added for improved security and better cost management. Unattended login sessions are less available to unauthorized parties. By closing all connections started by the login session unwanted connection costs are minimized.

Note that if you do not manually issue the t command with a different time value your login session will be terminated 900 seconds (i.e. 15 minutes) after the last user input (i.e. keystroke). The timer does not care what kind of application is running at the time or whether or not a data transfer is in progress, so in case you expect a long, non-interactive terminal session (setup tool monitoring, ISDN trace session, very large data transfer, etc.) you should disable the timer.

See The t Command on page 16 for information on using and changing the Auto-Logout timer.

OSPF Virtual Link Support

OSPF Virtual Links are now supported on the BRICK. Note that in OSPF the backbone, Area 0.0.0.0, is the center for all areas in the Autonomous System. However, sometimes it's not possible to physically connect all areas to the backbone. By configuring a "Virtual Link" between two area border routers a remote area can still be assigned to the backbone.

Virtual Links must be configured from the SNMP shell using the new ***ospfVirtIfTable***. See Configuring OSPF Virtual Links on page 16 for a detailed description/example for configuring virtual links.

Database Overflow

The maximum size of the OSPF Link State Database can now be controlled using the new *ipExtIfLsdbLimit* variable. This feature is intended for larger sites using OSPF where the Link State Database can become very large. For such sites some routers may not be able to handle (memory based limitations) the complete database.

See Controlling Link State Database Overflow on page 17 for detailed information on how overflow control works.

SetupTool Search Functionality

Setup Tool lists are now automatically sorted alphabetically and can be easily navigated using a new search mechanism. For information about how list searching works see the section Searching Menu Lists in Setup Tool on page 18.

This feature is mainly intended for sites with many partner interfaces.

Delayed Callback

The PPP Callback feature on the BRICK has been extended. An additional value (delayed) has been added for the answering side (see: Explanation of Terms) that delays the callback for the amount of time (in seconds) configured in the *RetryTime* variable of the ***biboPPPTable***.

The following table gives an overview of the Callback options currently available:

Setup Tool	SNMP Shell	Explanation
no	disabled	no Callback possible

Setup Tool	SNMP Shell	Explanation
expected (await-ing callback	expected	wait for a call back from a partner
yes	enabled	accept callback requests and call back immediately
yes (delayed)	delayed	accept callback requests and call back after *RetryTime* seconds
yes (PPP negotia-tion)	ppp_offered	accept callback requests and negotiate the callback number inband (see section New PPP Callback Method for details)

Note that *delayed* callback currently only works for calls identified outband by their CLID.

DNS Negotiation Configuration

DNS Negotiation can now be configured on a per partner basis allowing you to better control how (and from which partners) the BRICK will negotiate DNS settings.

Beginning in Release 4.6 Revision 2 each Dial-Up partner can be separately configured so that the BRICK either:

1. Accepts DNS settings from the partner.
2. Offers DNS settings to this partner.
3. Does not negotiate DNS settings with the partner.

See the section Partner-Specific DNS Negotiation on page 20 for information on configuring partner-specific DNS negotiation settings.

Extended DHCP Support

Some RFC 2131 based DHCP clients use DHCP messages that are not compatible with the RFC 1541 conformant DHCP Server on the BRICK. This means that the BRICK was unable to reject requests from DHCP clients on a different subnet than the BRICK. Problems occured e.g. when moving a laptop PC from a LAN with a Windows NT-based DHCP server to a different LAN with a BRICK as DHCP server—the laptop then did not accept the new IP address offered by the BRICK.

The BRICK now supports RFC 2131 based clients, solving the problem.

CAPI Extensions

CAPI 2.0 and 1.1

The CAPI_GET_VERSION and CAPI_GET_SERIAL_NUMBER messages can now be used under CAPI 1.1 and CAPI 2.0 to retrieve hardware and / or software information about the BRICK.

CAPI_GET_VERSION
: Returns the BRICK's current software version,
(retrieved from the ***biboAdmSWVersion*** object).

CAPI_GET_SERIAL_NUMBER
: Returns BRICK serial number information.
The information consists of a 7 character string that identifies the product type and SystemID as follows:

Digit 1: Identifies the product:

4 = BRICK-XS, BRICK-XS Office, V!CAS
5 = BRICK-XM
6 = BRICK-XL
7 = BinGO!

Digits 2-7: The last 6 digits of the BRICK's serial number.
(From the ***biboAdmSystemId*** object).

CAPI 2.0

The B2 protocol 12, "LAPD according to Q.921 including free SAPI selection." is now supported on the BRICK.

Bugfixes

TFTP put Transfers

- When transferring files to remote hosts via the TFTP “put” command signed variables were sometimes transmitted as unsigned. As of revision 2 this no longer occurs.

CAPI 1.1 and 2.0

- The BRICK sometimes rebooted when the CAPI and TAPI port numbers were exchanged or when large amounts of data were transmitted over a telnet session to the CAPI port. This has been corrected.
- When the ***isdnDispatchTable*** and ***isdnloginAllowTable*** are empty the BRICK’s CAPI server cannot distribute incoming calls because the isdnlogin service accepts the call. In previous releases CAPI applications were falsely notified of the call and then received a DISCONNECT_IND (because the call was already given to the isdnlogin service). CAPI applications are no longer notified of the call in such cases.
- When a CAPI application was started the message: “INVALID MESSAGE LENGTH!“ was improperly displayed in a running capitrace session. This problem has been corrected.

CAPI 2.0

- If during a fax transmission the remote side is no longer responding, the CAPI application is sent the error message “CAPI2_E_FAX_REMOTE_PROCEDURE_ERROR“ instead of a “CAPI2_E_FAX_REMOTE_ABORT“ message as in previous releases.
- In previous releases the CAPI 2.0 “FACILITY_REQ” message was not transmitted when the “Facility Request Parameter” field exceeded 6 bytes.

OSPF

- OSPF dialup links can now be safely used in active mode (Reference: Note in Release 4.6.1 on page 21.)

PPP

- As of release 4.6 Revision 2 the BRICK supports Cisco's proprietary (non RFC 1974 conformant) packet format for compressed (STAC) PPP datagrams. In some versions of Cisco software CCP negotiation is repeated periodically even when the connection and CCP has already been established. Prior to revision 2 this behaviour resulted in a memory leak on the BRICK.

RADIUS

- The Access Request ID is now changed for each new request (not including retransmissions) sent to the RADIUS server. In release 4.6.1 this ID wasn't changed for the second access request (e.g., when RADIUS outband authentication failed, RADIUS inband authentication followed with the same ID). This caused problems with some Radius implementations.

PMX

- The ***pmxIfErrors*** variable is now correctly adjusted upon unsuccessful B-channel initialization.

Detailed Feature Descriptions

The `t` Command

The **t** command has been added to the SNMP shell and defines the number of seconds to wait (once terminal input is idle) before closing the current login session. When the BRICK closes the login shell, all programs (setup session, trace, etc) started during the session that are currently running are also closed.

Usage:

Each time a user logs in the timeout is set to 900 seconds by default. To change the timeout setting enter:

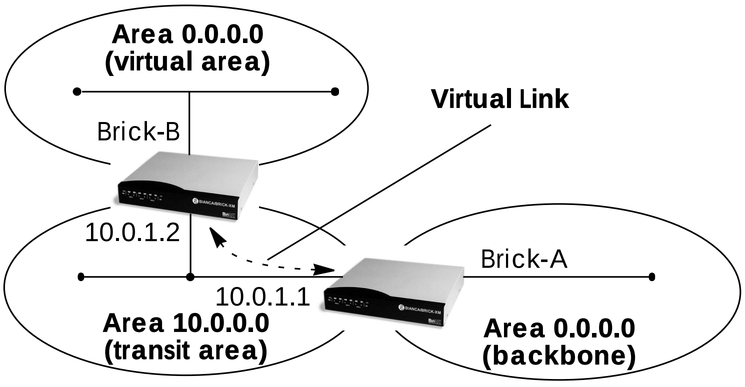
t <*seconds*>

The auto-logout feature can be disabled completely (for the current login session only) by setting the timer to 0.

NOTE: This feature is primarily intended for security/cost-control reasons. If you expect a long, non-interactive terminal session (setup tool monitoring, ISDN trace session, etc.) you should disable the timer.

Configuring OSPF Virtual Links

A virtual link is established between two Area Border Routers that share a common area; called the “transit area”. Both routers must be physically connected to the backbone.



A virtual interface must be defined on each of the ABRs by creating an entry in the ***ospfVirtIfTable***. This is done by setting the ***ospfVirtIfNeighbor*** and ***ospfVirtIfAreaID*** objects.

ospfVirtIfNeighbor should be set to the Router ID of the Area Border Router at the oher end of the virtual link.

ospfVirtIfAreaID should be set to the area ID of the transit area.

The virtual link in the diagram above would be configured on Brick-A as follows.

```
BRICK-A:system> ospfVirtIfTable

inx  AreaId(*rw)           Neighbor(*rw)           TransitDelay(rw)
     RetransInterval(rw)   HelloInterval(rw)       RtrDeadInterval(rw)
     State(ro)             Events(ro)              AuthKey(rw)
     Status(-rw)           AuthType(rw)

BRICK-A:ospdVirtIfTable> AreaID=10.0.0.0 Neighbor=10.0.1.2
```

This creates a new OSPF virtual interface (on BRICK-A) that links two parts of the backbone via the transit area 10.0.0.0. The respective interface would be created on BRICK-B using almost the same command (***ospfVirtIfAreaID***=10.0.0.0 ***ospfVirtIf-Neighbor***=10.0.1.1)

Remember that the area being used as the transit area must already be defined in the ***ospfAreaTable***.

Controlling Link State Database Overflow

Sites with large (or complicated) network installations that are running OSPF may notice the Link State Database (LSDB) becoming large. Most often this is the case where external routes are being imported as external advertisements.

One way to minimize the size of the LSDB (on the BRICK) is to use the ***ospfExtLsdbLimit*** variable. This object defines the maximum number of external LSAs to store in the database (the local copy).

Once the limit is reached the BRICK goes into Overflow State. In Overflow State two things happen:

1. The BRICK begins to flush all external advertisements generated locally.
2. The BRICK ignores all new external advertisements.

Note: The maximum size of the LSDB must be the same for all OSPF routers in the domain for this feature to perform efficiently.

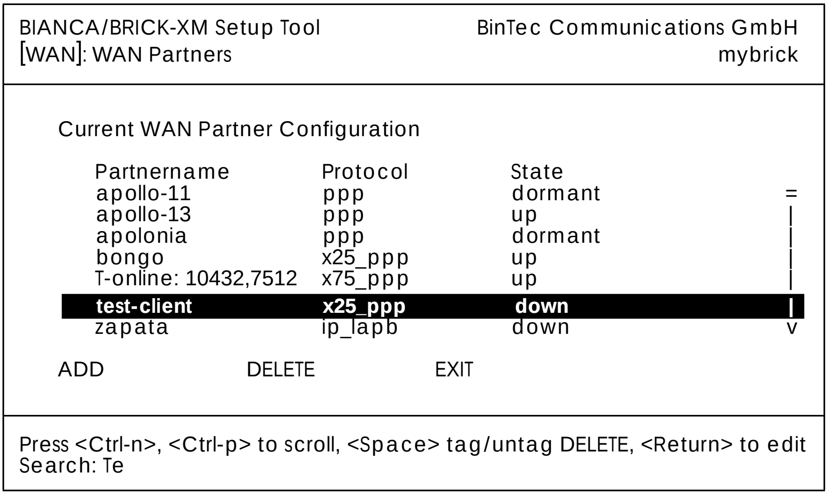
By default the BRICK remains in overflow state but can optionally be configured to leave overflow state (and continue to process new external LSAs) automatically after a time period. The ***ospfExtOverflowInterval*** variable defines the number of seconds to wait before leaving overflow state automatically. The default is 0 seconds (i.e., stay in overflow state). After waiting ***ospfExtOverflowInterval*** seconds the number of external LSAs in the LSDB is compared to the ***ospfExtLsdbLimit***. If there is room in the database for new LSAs the BRICK then leaves overflow state; otherwise another time interval is waited.

The diagram shown below attempts to illustrate the behavior of database overflow control using the ***ospfExtLsdbLimit*** and ***ospfExtOverflowInterval*** variables.

Searching Menu Lists in Setup Tool

Setup Tool menus that display list entries are sorted alphabetically using the contents of the first field.

To search menu list items enter a valid search character (only printable characters). The cursor automatically jumps to the first match in the list. As long as the search is active subsequent characters entered are appended to the search string. The current search string is shown in the bottom portion of the terminal window. Entering a non-printable character resets the current search (and possibly performs an action; e.g. tab, space, etc.). The <backspace> key (and possibly <delete> depending on terminal settings) can be used to edit the search string. Search characters are case-insensitive (Entering the letter "t" matches both "t" and "T" characters).



```
BIANCA/BRICK-XM Setup Tool                         BinTec Communications GmbH
[WAN]: WAN Partners                                                   mybrick

   Current WAN Partner Configuration

      Partnername            Protocol          State
      apollo-11              ppp               dormant                 =
      apollo-13              ppp               up                      |
      apolonia               ppp               dormant                 |
      bongo                  x25_ppp           up                      |
      T-online: 10432,7512   x75_ppp           up                      |
      test-client            x25_ppp           down                    |
      zapata                 ip_lapb           down                    v

   ADD              DELETE            EXIT

Press <Ctrl-n>, <Ctrl-p> to scroll, <Space> tag/untag DELETE, <Return> to edit
Search: Te
```

Assuming the above [WAN Partners] menu list the following key sequences would have the following effect:

Key Sequence	**Resulting Effect**
t, or **T**	Cursor jumps to the: **T-Online 10432,7512** entry.
te, **TE**, **tE**, **Te**	Cursor jumps to the: **test-client** entry.
a p o l o	Cursor jumps to: **apollo-11** entry first then to: **apolonia** after the last "o".

Note also that a search can only be performed when the cursor is in a list field (and not when in an ADD, DELETE, EXIT, CANCEL, or SAVE field).

Partner-Specific DNS Negotiation

Together the MIB variables ***biboPPPIpAddress*** and ***biboPPPDNSNegotiation*** control how DNS negotiation is handled with the respective PPP partner.

biboPPPDNSNegotiation

The type of negotiation to perform with this client.
Default value: “enabled”. Possible values include:

`disabled`(1), `enabled`(2),
`dynamic_client`(3), `dynamic_server(4)`

biboPPPIpAddress

The type of IP address for this dial-up partner.
Possible values include:

`static`(1), `dynamic_server`(2), `dynamic_client`(3)

The table below illustrates the effect of using theses two variables to control DNS negotiation.

Variable:Setting:	**Negotiation Handling:**
Negotiation=`disabled`	No negotiation is performed.
Negotiation=`enabled` **AND** ***IpAddress***=`dynamic_client`	Remote side is always asked for primary/secondary (***biboAdmNameServer***/***biboAdmNameServ2***) server. Local values are always overwritten is offered by remote.
Negotiation=`enabled` **AND** ***IpAddress***=`dynamic_server` **OR** ***IpAddress***=`static`	Values for primary/secondary server are only sent if requested by remote side and they are configured.
Negotiation=`dynamic_client`	Remote side is always asked for primary/secondary nameserver addresses.
Negotiation=`dynamic_server`	Values for primary/secondary servers are sent only when requested by remote side.

Release 4.6 Revision 1:

Released: 01.10.97

Features: | Bugfixes: | Detailed Description:

Features

New Routing Protocol OSPF

With release 4.6.1 the BRICK-XM is now capable of using the OSPF (Open Shortest Path First) routing protocol. This protocol is more efficient than the standard RIP protocol, especially for larger networks.

Note: ⚠ You will need a new license key to use OSPF on your BRICK. New license keys for your BRICK-XM can be purchased from your distributor or directly from BinTec Communications.

For a detailed introduction to OSPF and its configuration please refer to page 43.

Note: ⚠ With release 4.6.1 you should only use OSPF in passive mode for ISDN interfaces, unless the ISDN interface belongs to the same OSPF area as the active LAN interface.

New charge-dependent Short Hold

You can now configure a charge-dependent Short Hold on your BRICK.

For a detailed description of this feature and its configuration please refer to page 31.

New PPP Callback Method

With release 4.6.1 an additional Callback method according to RFC 1570, named *ppp_offered*, is available.

For a detailed description of this feature and its configuration please refer to page 34.

New enhanced RADIUS features

The BRICK’s RADIUS features have been improved.

For a detailed description of these features and their configuration please refer to page 38.

BIANCA/CM-PRI

You can now create SNMP traps which are triggered by changes of the *pmxIfLayer1State*.

BRICKware for Windows

- The »Reflection« TCP / IP stack for Windows from WRQ is now also supported by the BinTec CAPI2032.DLL.
- The ISDN trace of Dime Tools now also decodes LCP extensions and IPCP extensions.
- Improved Time Server features; The Dime Tools Time Server is now also capable of supplying your BRICK with its local time.

  You can now choose between Greenwich Mean Time (GMT) and Local Time in the Configuration–Time Server menu of Dime Tools.

  We recommend that you use the Local Time setting of Dime Tools and set Time Offset (seconds) to 0 in the BRICK’s [*IP*][*Static Settings*] Setup Tool menu. This will also automatically take care of daylight savings time[1].

  Note, however, that this method is no longer fully compatible to RFC 738.

  If you select Greenwich Mean Time, the current setting for the TZ (timezone) environment variable in the AUTOEXEC.BAT will be displayed on the screen. For Germany this will usually be

  set TZ=GST1GDT

1. Called »Sommerzeit« in Germany.

Please refer to your PC documentation for a description of the TZ entry.

DHCP Server

When using dynamic IP address assignment you can now associate an IP address with a MAC (hardware) address. This can be useful for small networks which shall be managed from a central server.

In Setup Tool this can be done in the [*IP*] [*DHCP Server*] [*ADD*] menu. Set the *Number of consecutive addresses* to **1** and enter the appropriate *MAC Address*.

ifstat Command

The *ifstat* command now takes the following three optional parameters:

-*l*	Displays the full length of the interface descriptions (normally the description is only displayed up to the 12th character).
-*u*	Only display information on interfaces which are in the **up** state.
‹ifcname›	Only display information on interfaces whose description starts with the given characters (e.g. **ifstat en1** will display information on the interfaces en1, en1-llc, and en1-snap).

IP

New ipPriorityTable

We added the ***ipPriorityTable*** to the MIB to be able to make routing protocol priorities user-definable.

This can be useful when you want your BRICK to prefer certain routing protocols over others.

Each table entry consists of two variable, Proto, which specifies the routing protocol, and Value, which specifies the priority for this protocol. Value can be 0 to 63, where lower numbers define a higher priority, i.e. 0 is the highest priority, 63 the lowest.

The available routes with the highest priorities are always used when routing IP packets.

The following route types are supported at the moment:

direct	Routes where *ipRouteType*=**direct**, which define IP addresses for interfaces. Default priority: 0
static	Routes where *ipRouteType*=**indirect**, which were not generated by a routing protocol. Default priority: 0
ospf	OSPF intra area or inter area routes. Default priority: 10
ospf_ext	OSPF external routes. Default priority: 15
rip	Routes which were learned by the RIP protocol. Default priority: 20

New ipExtIfTcpCksum Switch

The ***ipExtIfTable*** now contains the variable *ipExtIfTcpCksum* which can be used to enable (**check**) or disable (**dont_check**) TCP checksumming.

We recommend setting this switch to **dont_check** only on LAN interfaces for performance critical CAPI applications.

Do *not* use **dont_check** for interfaces where Van Jacobson Header Compression is enabled, or over which other routers can be reached, because in these cases the correct function of the TCP cannot be guaranteed.

For traffic inside your LAN you can disable (**dont_check**) the TCP checksum, because ethernet and token ring have their own CRC checksums to ensure data integrity.

IP Performance Optimization

IP performance was optimized, especially for Remote CAPI applications.

IP Routing

While setting up a dialup interface (ISDN) for IP traffic other routes to the same IP destination are used until the dialup interface has reached the **up** state. This ensures a smooth transition from an alternative route back to the main route.

In previous releases IP traffic was blocked for the duration of the interface setup.

ISDN

- A Syslog message is now created each time the isdnlogind takes an incoming call.
- A new variable, *DialOutPrefix*, was added to the ***isdnStkTable***. The contents of this variable are appended to the callback number when using PPP inband callback (Microsoft CBCP, see page 36).

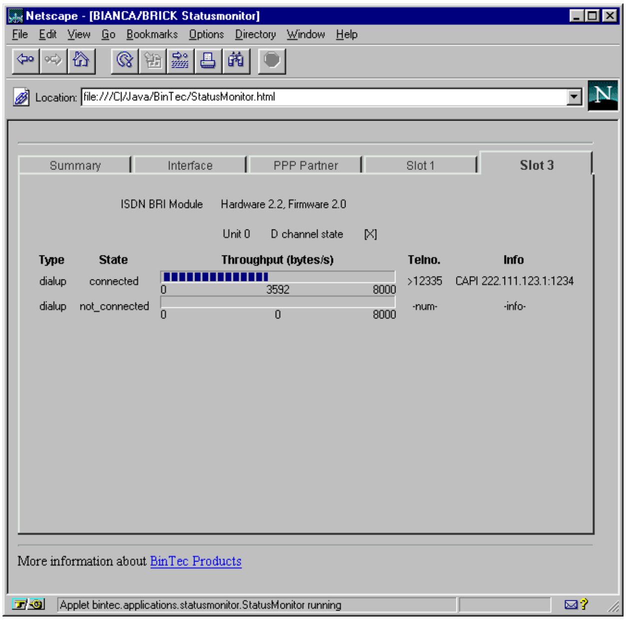
New Java Status Monitor

The Java Status Monitor allows you to monitor your BRICK router dynamically and see:

- how long the system has been running
- current traffic for each interface
- connection information (which partners are connected, duration of call, accumulated costs, ...)
- number of B-channels currently in use, and details for each connection

You can choose from the BinTec ISDN Companion CD setup whether you want to install the Java Status Monitor.

For further information please refer to the STATMON.TXT and DEVELOP.TXT files installed with the Java Status Monitor.


Netscape - [BIANCA/BRICK Statusmonitor]
File Edit View Go Bookmarks Options Directory Window Help
Location: file:///C|/Java/BinTec/StatusMonitor.html
Summary
Interface
PPP Partner
Slot 1
Slot 3

ISDN BRI Module Hardware 2.2, Firmware 2.0

Unit 0 D channel state [X]

Type	State	Throughput (bytes/s)	Telno.	Info
dialup	connected	0 3592 8000	>12335	CAPI 222.111.123.1:1234
dialup	not_connected	0 0 8000	-num-	-info-

More information about BinTec Products

Applet bintec.applications.statusmonitor.StatusMonitor running

You can also download the latest version of these files, along with the complete Java Status Monitor package from our file server in the *ftp://ftp.bintec.de/pub/brick/statusmon* directory.

PPP

Local PPP ID for each WAN Partner

You can now specify a different Local PPP ID for each WAN partner. This ID is stored in the *biboPPPLocalIdent* variable in the ***biboPPPTable***. If you do not specify a special Local PPP ID for a partner, the default PPP ID is taken from the *biboAdmLocal-PPPIdent* variable in the ***admin*** table.

In the Setup Tool you can configure the Local PPP ID field in the [*WAN Partner*][*ADD*] menu.

MTU Negotiation

The remote MRU / MRRU (Maximum Receive Unit, maximum packet length) negotiated by the LCP (Link Control Protocol) will now be written to the *ifMtu* variable of the corresponding ***ifTable*** entry to avoid packet discarding by peer caused by exceeding the negotiated peer's MRU / MRRU.

RIP

RIP performance was improved. DEBUG level syslog messages are now also generated when routes are created or deleted.

Setup Tool

DHCP Server

You can now specify a MAC address when configuring your BRICK as a dynamic IP address server (see section DHCP Server above).

Local PPP ID

You can now specify a different Local PPP ID for PAP / CHAP authentication for each WAN Partner (menu [*WAN Partner*] [*ADD*]).

X.25

In the [*X.25*][*Link Configuration*][*ADD*] menu you can now specify *default* and *maximum* values for the *Packet Size* and *Window Size*. The *Packet Size* entries can be 128 to 4096 bytes, the *Window size* entries can be 2 to 127.

System

At each system start a syslog message will now be generated.

"system ‹sysname› started at ‹time›"

Bugfixes

BRICKware for Windows

- Remote CAPI: In previous releases when you installed the remote CAPI as the Administrator on Windows NT systems, you could not access them as a normal user, because of insufficient access permissions.
  The access permissions have been changed to allow all users of the system to use the remote CAPI.

CAPI

- You can now set up X.75 connections over V.110 or modem. Asynchronous HDLC framing is used for these connections.
- When using Direct Dial In (DDI) on point-to-point accesses[1] the *SendingComplete* D channel message was mistakenly acknowledged with *SetupAcknowledge*, thus preventing the call from being accepted.
  This bug has been fixed.

CAPI 2.0

- CAPI 2.0 applications could not evaluate the ISDN subaddress of incoming calls.
- The BRICK did not send any INFO_IND messages (e.g. charging information, direct dialling in digits) to CAPI 2.0 applications.

These two bugs have been fixed.

IP

- Packets with a length of MTU+1, MTU+2, and MTU+3 (where MTU is the maximum transfer unit length) were mistakenly discarded in previous releases.
  They are now correctly handled.

1. Called »*Anlagenanschluß*« in Germany

ISDN

- When using the *AT&T 5ESS Custom ISDN* or *National ISDN 1 Northern Telecom DMS100* protocols now µ-law is signalled for CAPI (CIP Translation) as required.
- BIANCA / CM-PRI: Short Layer 1 dropouts during high system load periods occasionally led to a blocked PMX interface, i.e. no new calls could be set up.
  This bug has been fixed.

PPP

- x75_ppp encapsulation: Asynchronous PPP is now used as a default for both incoming and outgoing connections, it is no longer dependent on a configured *PPPLoginString*. The logon procedure is an exception to this rule. During this phase no asynchronously encapsulated packets will be sent or received.
- Inband CHAP authentication with Cisco equipment now works correctly if you configure CHAP for both directions.

RIP

- There was a problem when using different IP network addresses on the same physical network—subnet routes were mistakenly taken to be host routes.
  This problem has been fixed.
- Default routes can now be learned using RIP.

Setup Tool

- When configuring a BIANCA / CM-PRI interface as a Hyperchannel now the correct ISDN channel (0) is set to *leased_dte*.

SNMP

- The *ifNumber* variable in the ***interfaces*** table is now set correctly.

X.25

- When you changed the *Speed* value in the ***X21IfTable***, a second ***X25LinkTable*** entry was mistakenly created. One entry was set to *State* **ready**, while the other remained at *State* **sabm_pending**. This led to problems, because the BRICK then waited for a RESTART CONFIRMATION, which never arrived.
  This error no longer occurs.
- *NSAP and »Marker« Facility*: When the BRICK received a CALL REQUEST with NSAPs which was deleted by an ***x25RewriteTable*** entry, the corresponding Marker facilities were mistakenly not deleted.
  The BRICK then sent CALL REQUESTs with Marker facilities, but without NSAPs.
  This error no longer occurs.

Detailed Feature Descriptions

Charge-dependent Short Hold

The Short Hold timer available in previous releases could be used to disconnect a dialup call after a configurable, but fixed, period of inactivity.

As ISDN calls are normally not charged according to the exact length of the connection in seconds, but rather according to a coarser grid of charging units—which can be anything from a few seconds to several minutes in length, depending on the target you are calling, the time of day, etc.—the fixed solution mentioned above is not flexible enough to adapt the Short Hold timer to the changing charging unit lengths.

With release 4.6.1 you can configure your BRICK to adapt the short hold timer dynamically depending on the actual lengths of the call charge units (*Dynamic Short Hold*).

To be able to use the Dynamic Short Hold your ISDN access must have the AOCD (advice of charge during the call[1]) feature activated.
If you are not sure whether AOCD is activated for your ISDN access, there is an easy way to verify it.
Go to the [*Monitoring and Debugging*][*ISDN Monitor*] menu of the Setup Tool while an outgoing ISDN call is active. If the *Charge* field for this call remains empty until the end of the call, no advice of charge was received during the call.

New MIB Variables

Two new MIB variables were introduced to enable Dynamic Short Hold.

- The new *ChargeInterval* variable in the ***biboPPPStatTable*** contains the length of the last charging unit. It is updated every time a new advice of charge is received.

1. Called *»Übermittlung der Tarifeinheiten* während *der Verbindung«* in Germany

The *ChargeInterval* variable is reset to 0 if no advice of charge is received for an hour.

The variable will remain at 0 if the AOCD feature is disabled, or not available from the network or PBX.

- The new *DynShortHold* variable of the ***biboPPPTable*** specifies the Dynamic Short Hold idle timer as a percentage of the current *ChargeInterval*.

  For example, if you set the *DynShortHold* variable to 75 (%), and the last measured *ChargeInterval* was 120 seconds, the idle timer will be set to 90 seconds. As soon as the *ChargeInterval* length changes, the idle timer setting will change accordingly.

Configuring Dynamic Short Hold

Static Short Hold is configured as before. Dynamic Short Hold is activated by specifying a percentage of the charge unit length (*ChargeInterval*) either in the [*WAN Partner*][*ADD*][*Advanced Settings*] menu of the Setup Tool, or in the *DynShortHold* variable of the ***biboPPPTable***.

As a default, Dynamic Short Hold is *not* active (0%).

- For *interactive connections* (e.g. telnet) you should specify a rather high Dynamic Short Hold percentage (e.g. 80-90) to avoid frequent disconnects due to short periods of inactivity.
- For *internet connections* (WWW, http, etc.) you should specify a medium to high Dynamic Short Hold percentage (e.g. 50-80) to avoid frequent disconnects due to waiting periods.
- For *data connections* (e.g. ftp) you should specify a low Dynamic Short Hold percentage (e.g. 10-40) to avoid unnecessarily waiting—and incurring charges—once a transfer is complete.

Note: ⚠ If configured, the Static Short Hold timer will *always* take precedence over Dynamic Short Hold to avoid permanent connections.
Make sure to set the Static Short Hold to a value greater than the length of a charging unit if you want Dynamic Short Hold to have any effect.
For example, in Germany there are different maximum charging unit lengths for different tariff zones (City = 4 minutes, long distance calls = 2 minutes), so you can set the *Static* Short Hold to 245 (>4 minutes) for City connections, and to 125 (>2 minutes) for long distance calls, to avoid nullifying your Dynamic Short Hold settings.

Once the Dynamic Short Hold inactivity time is reached, the connection will be kept up until shortly before the next advice of charge is expected, thus maximizing the connection time without any additional cost.

This mechanism will not work properly for the first charging unit with a radically changed length once a new tariff zone is entered, which may result in a few inefficiently used longer charging units.

☞ If you are using Dynamic Short Hold in connection with channel bundling, please note that the channels are released one by one, keeping open each channel until short before the next advice of charge is expected for this channel, thus maximizing the connection time without further cost.
The call will of course be disconnected immediately if either side actively closes it.

Syslog Messages

Syslog messages are created when Dynamic Short Hold disconnects a call.

```
DEBUG/PPP: dialup1: dynamic shorthold, 89 seconds after last
advice of charge
```

```
INFO/PPP: dialup1: outgoing link closed, duration 180 sec,
1950 bytes received, 1946 bytes sent, 2 charging units
```

The messages above tell you that Dynamic Short Hold disconnected the call, and supply some connection statistics.

```
DEBUG/PPP: dialup1: shorthold timeout reached

INFO/PPP: dialup1: outgoing link closed, duration 63 sec, 438 bytes
received,  434 bytes sent, 1 charging units, no AOCD
```

These messages give an example of a call disconnected by the Static Short Hold. They also tell you, that no advice of charge packets were received.

New PPP Callback Method

Overview

With release 4.6.1 there is now an additional Callback method, named **ppp_offered**. This method can be used to negotiate Callback and the number to use in the LCP—according to the LCP extensions specified in RFC 1570.

This solutions is also very flexible, because you do not need to modify the configuration at the central site each time the »sales representative« moves on to a new location—the callback number is newly transmitted at each call.

The **ppp_offered** callback has to be enabled for each partner explicitly, because it is a potential security loophole.

Explanation of Terms

In the following we will always use the term »caller« to denote the person who initially calls the central site and wishes to be called back, and the term »answerer« to denote the central site.

MIB Changes

To support the new callback method we made a few necessary changes to the MIB. The variables *biboPPPCallback* and *biboDialType* now take new values, and the ***isdnStkTable*** contains the new variable *DialOutPrefix*.

1. biboPPPCallback

biboPPPCallback now also can be set to **ppp_offered**, in addition to **enabled**, **disabled**, and **expected**, as before.

ppp_offered enables the LCP callback negotiation on the answerer side. The answerer then acknowledges the callback number sent by the caller via LCP and creates a temporary entry in the ***biboDialTable***. This entry contains the callback *Number* sent by the caller, *Type* is set to **ppp_negotiated** (see 2. below), *Direction* is set to **outgoing**, *Screening* is set to **dont_care**, *IfIndex* is set to the appropriate interface.

The temporary entry is deleted once the callback connection is closed.

When initiating the actual call back, this entry is used, unless there is an entry with the same *IfIndex*, where *Type* is **isdn** or **isdn_spv**, and *Direction* is **both** or **outgoing** (see 2. below).

Entries with *Type* **isdn** and **isdn_spv** take precedence over entries with *Type* **ppp_negotiated**.

If any or all ISDN interfaces of the answerer are connected to the ISDN via a PABX and you have to dial a prefix code for external calls these prefix codes must be configured in the variable *DialOutPrefix* in the ***isdnStkTable*** (see 3. below).

2. biboDialType

biboDialType now also can be set to **ppp_callback** and **ppp_negotiated**, in addition to **isdn**, **isdn_spv**, and **delete**, as before.

Type **ppp_negotiated** is only used by the answerer to mark the temporary entries created by an LCP callback negotiation.

Type **ppp_callback** is only used by the caller to mark entries which contain the callback number for an LCP callback negotiation.

3. isdnStkDialOutPrefix

This new variable in the *isdnStkTable* must be used to configure prefix codes if any or all of the answerer's ISDN interfaces are connected to the ISDN via a PABX. In many cases this is simply the digit »0«.

This prefix is then dialled before the callback number supplied by the caller.

☞ At the moment this variable is exclusively used for callbacks initiated via an LCP callback negotiation.

4. Microsoft Extension CBCP (CallBack Control Protocol)

The LCP callback negotiation as specified in RFC 1570 has a few inherent disadvantages, e.g. when using inband authentication the callback number supplied by the caller is acknowledged by the answerer *before* identifying the caller.

Microsoft proposed CBCP as an internet draft (»Proposal for CallBack Control Protocol (CBCP)«, July 19, 1994) to remedy some of these disadvantages.

Microsoft software uses CBCP for connections made via Dial-Up Networking as well as via Remote Access Service (RAS) Server / Client.

As a default LCP callback negotiation as specified in RFC 1570 is used for connections from BRICK to BRICK. When communicating with Windows systems CBCP is also accepted.

5.a Configuring Callback on the BRICK

Create a WAN partner entry with Authentication set to CHAP (in Setup Tool set the *PPP Authentication Protocol* to **CHAP** in the [*WAN Partner*][*ADD*] menu), and *Callback* set to **ppp_offered** (in Setup Tool set the *Callback* to **offered** in the [*WAN Partner*][*ADD*][*Advanced Settings*] menu).

Make sure that *Identify by Calling Number* is set to **no** for modem call-ins. Fill in the other fields as appropriate for this partner.

5.b Configuring Callback under Windows 95

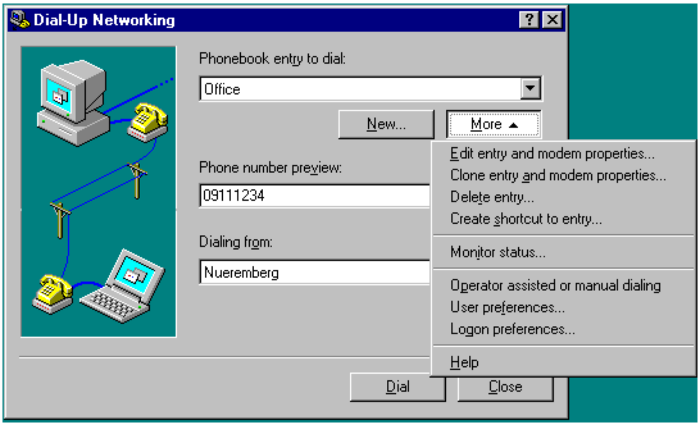
Callback is not explicitly configured under Windows 95. Use your normal Dial-Up Networking settings.

During connection setup the BRICK offers Callback, the PC then opens a dialogue box where the user can enter the number to use for this callback. Once this has happened the connection is cleared, and the BRICK initiates the callback.

Note: The initial connection will remain open until the user has entered a callback number.

5.c Configuring Callback under Windows NT

Under Windows NT[1] go to Dial-Up Networking and click on the »More« button.



Dial-Up Networking

Phonebook entry to dial: Office

Phone number preview: 09111234

Dialing from: Nueremberg

New...
More
Edit entry and modem properties...
Clone entry and modem properties...
Delete entry...
Create shortcut to entry...
Monitor status...
Operator assisted or manual dialing
User preferences...
Logon preferences...
Help
Dial
Close

Then select »*Edit entry and modem properties…*«.

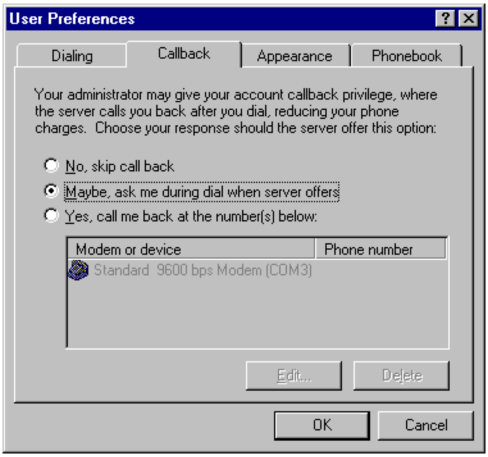
Configure the basic settings for this entry, on the Server page select the Dial-up server type »*PPP: Windows NT, Windows 95 Plus, Internet*«, select TCP / IP as a network protocol, and *Enable PPP LCP extensions*.

On the Security page select »Accept only encrypted authentication«.

Confirm your settings with »OK«.

Finally, click on »More« once again and select »*User preferences…*«. There you can configure callback.

1. This example is for Windows NT Client 4.0, SVP 3, the dialogues may look slightly different on your system.


User Preferences

Dialing | Callback | Appearance | Phonebook

Your administrator may give your account callback privilege, where the server calls you back after you dial, reducing your phone charges. Choose your response should the server offer this option:

- ○ No, skip call back
- ● Maybe, ask me during dial when server offers
- ○ Yes, call me back at the number(s) below:

Modem or device	Phone number
Standard 9600 bps Modem (COM3)	

Edit... Delete

OK Cancel

Choose »Maybe«. The PC will then ask you to specify the call-back number, as described above for Windows 95.

Note: With release 4.6.1 it is not yet possible to use the BRICK as a caller, when the answerer is a Windows NT RAS server. The BRICK can then only act as the answerer.

RADIUS

The BRICK's RADIUS features have been improved.

Standard RADIUS Attributes

Release 4.6.1 supports more of the standard RADIUS attributes than previous releases. Also a couple of BinTec-specific options have been added, to facilitate using your BRICK in conjunction with a RADIUS server.

Note, however, that the BinTec-specific options are only available if you use the *dictionary* file included on the Companion CD (the file is also available from our WWW server).

The following standard RADIUS attributes are available.

New or enhanced attributes are set in an *italic* typeface.

RADIUS Attribute	**Type**	**R / A**	**Remark**
User-Name	string	REQ	User name, mandatory inband: PPP partner name *outband: PPP partner telephone number*
User-Password	string	REQ	Password for PAP authentication
CHAP-Password	string	REQ	Password for CHAP authentication
NAS-Identifier	string	REQ	sysName of the BRICK
Service-Type	integer	ANS	Framed (for PPP) *Callback-Framed* (for PPP with Callback)
Framed-Protocol	integer	ANS	inband: PPP *outband:* *PPP, X25, X25-PPP, IP-HDLC, IP-LAPB, MPR-LAPB MPR-HDLC, FRAME-RELAY, X31-BCHAN, X75-PPP, X75BTX-PPP, X25-NOSIG, X25-PPP-OPT*
Framed-IP-Address	ipaddr	ANS	Partner IP address
Framed-IP-Netmask	ipaddr	ANS	Partner IP netmask
Framed-Routing	integer	ANS	None, RIPv1-Broadcast, RIPv1-Listen, RIPv1-Broadcast-Listen
Framed-Compression	integer	ANS	None, *Van-Jacobson-TCP-IP*
Framed-Route	string	ANS	You can create a route of the format ‹ipaddr›[/‹netmask bits›] ‹gateway› [‹metric1›...‹metric5›] e.g.: 192.2.3.4/24 193.141.54.1 1
Idle-Timeout	integer	ANS	Shorthold
Port-Limit	integer	ANS	Number of B channels (== MaxConn)
Reply-Message	string	ANS	*outband:* ifDescr *is set to this name (instead of using the telephone number)*
Callback-Number	string	ANS	*telephone number for Callback*

The following RADIUS attributes are *not* yet applicable to your BRICK:

Acct-Authentic	CHAP-Challenge	Login-IP-Host

Acct-Delay-Time	Callback-Id	Login-LAT-Group
Acct-Input-Octets	Called-Station-Id	Login-LAT-Node
Acct-Input-Packets	Calling-Station-Id	Login-LAT-Service
Acct-Output-Octets	Filter-Id	Login-Port
Acct-Output-Packets	Framed-AppleTalk-Link	Login-Service
Acct-Session-Id	Framed-AppleTalk-Network	NAS-Port-Type
Acct-Session-Time	Framed-AppleTalk-Zone	Proxy-State
Acct-Status-Type	Framed-IPX-Network	Session-Timeout
Acct-Terminate-Cause	Framed-MTU	Termination-Action
		Vendor-Specific

BinTec Extensions

If you use the dictionary file mentioned above you can directly access and configure specific MIB tables via RADIUS.

The following options are available at the moment:

Option	**Type**	**Mode**
BinTec-biboPPPTable	string	static
BinTec-ipExtIfTable	string	static
BinTec-ipRouteTable	string	dynamic
BinTec-ipExtRtTable	string	dynamic
BinTec-biboDialTable	string	dynamic
BinTec-ipNatPresetTable	string	dynamic

Each of these options corresponds to a MIB table. You can modify values inside the table by using a syntax similar to the SNMP client shell of your BRICK:

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

A few lines from a RADIUS setup file might look like this:

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

When using these options please note:

- The *ifIndex* is automatically set for each table, you cannot influence it.
  There is, however, one exception to this rule: In the ***IpExtRtTable*** both the *DstIfIndex* and the *SrcIfIndex* are automatically set. You can set one of these to 0 if need be.
- The entries are not case-sensitive.
- You must not use blank spaces before or after »=« signs inside the double quotes.
- There are two different option modes, static, and dynamic.

  Static options modify existing table entries while dynamic options add a new table entry. Therefore all the variables you want to set in a dynamic option have to be included in one single line.

Partner Recognition via CLID

To identify RADIUS partners outband by their CLID (calling line identification, i.e. ISDN telephone number) there has to be a corresponding entry in the RADIUS database, e.g.

```
9119732123    Service-Type = Framed,
              Framed-Protocol = IP-HDLC,
              Reply-Message = "partner1"
```

Note that the phone number must be specified here exactly as it is signalled with the incoming call (you can see this in the *RemoteNumber* field of the ***isdnCallTable***).

When a call from this number comes in a new PPP entry is generated with *Encapsulation*=**ip_hdlc** and *ifDescr*=**partner1**.

Please also note that when using RADIUS inband authentication it can take up to 2 seconds to accept an incoming call if the RADIUS server is delayed inactive.

At the moment it is not possible to use both inband and outband RADIUS authentication at the same time for one connection.

Channel Bundling

You can now bundle several B channels to achieve a higher data throughput using the *Port-Limit* option.

Note: ⚠ Certain RADIUS servers handle the setup of further B channels for a connection incorrectly.
This can result in very high charges!
So before using channel bundling for RADIUS make sure your RADIUS server is capable of handling it correctly.

RADIUS Table Entries

The *ifIndexes* of RADIUS PPP entries now start at 15001. They are not stored when saving your configuration.

Default RADIUS UDP Port

The default UDP port used for RADIUS authentication is 1645.

OSPF

OSPF (Open Shortest Path First), an interior routing protocol that can be used as an alternative to RIP, is now supported on all BRICK products. To use OSPF on the BRICK a supplemental license is required.

Because the OSPF protocol is generally more complex and is not as well known as RIP this document is broken down into the following sections.

1. A general description of the OSPF protocol (p. 43)
2. System table reference for the OSPF subsystem (p. 55).
3. Setup Tool menus for configuring OSPF (p. 58).
4. An example OSPF installation showing the required configuration steps using Setup Tool (p. 67).
5. Troubleshooting OSPF (p. 76).

The OSPF Protocol

OSPF (Open Shortest Path First), is an interior routing protocol that is often used by larger network installations as an alternative to RIP. It was originally designed to address some of the limitations of RIP (when used in larger networks). Some of the problems (with RIP) that OSPF addresses include:

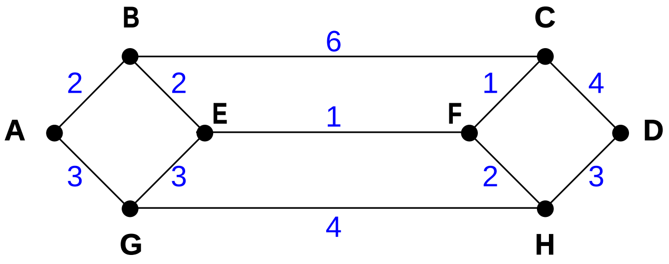
- **Faster Network Convergence**
  Changes in routing information are propagated immediately when changes occur and not periodically as with RIP.
- **Reduced Network Load**
  After a brief initialisation phase, routing information does not need to be refreshed as in RIP where the entire routing table is broadcast every 30 seconds.
- **Routing Authentication**
  Routers advertising OSPF routes can be authenticated.
- **Routing Traffic Control**
  OSPF areas can be closed to limit the amount of traffic resulting from routing advertisements.

- **Link-Costs**
  When calculating a route's cost OSPF can account for the different transport mediums such as LAN or WAN links.
- **No Hop-Count Limitations**
  In RIP, routes spanning more than 15 hops are unreachable.

Although the OSPF protocol is more complex than RIP the basic concept is the same; the best interface must be calculated for forwarding packets to a particular station.

Shortest Path Routing

With RIP, routes are measured and selected according to number of hops it takes for a packet reach it's destination. In the diagram below, each node represents an IP router. According to RIP, the best route for a packet travelling from A to C will always be ABC.



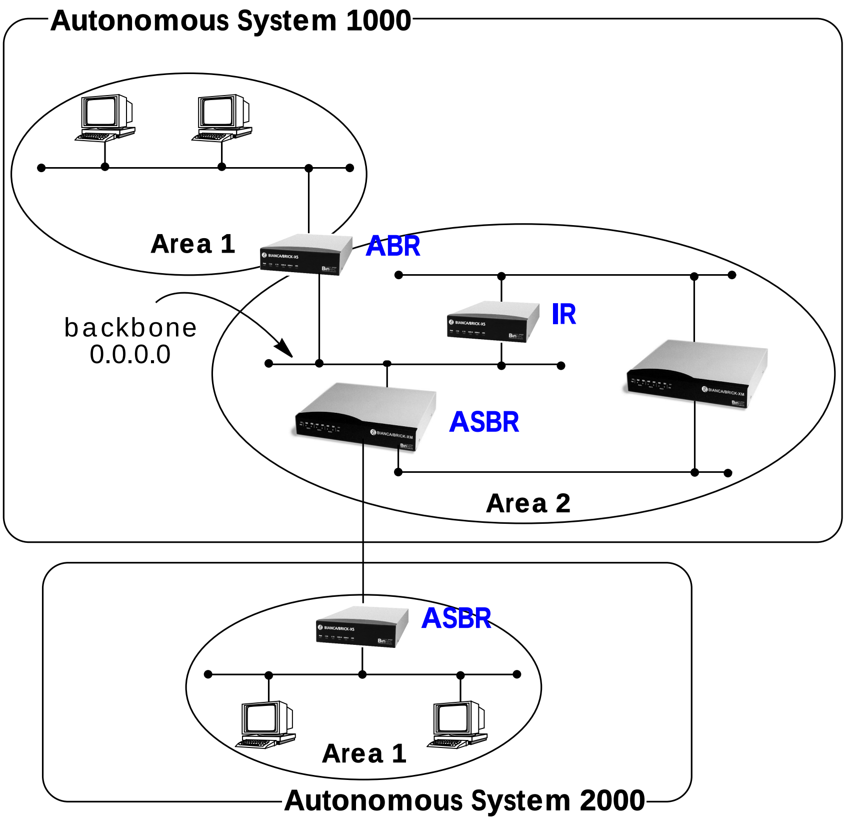
In OSPF each link has a cost associated with it (typically some fixed number divided by the bandwidth of the link). Routes are calculated and selected according to the least cost of the overall path a packet will travel. Thus in shortest-path routing the best path is also the fastest path (theoretically), regardless of the number of stations a packet travels through.

Assuming the relative costs of the links in the diagram above (shown in blue), according to OSPF the best route for a packet travelling from A to C is ABEFC (cost = 6). This route requires 4 hops as opposed to the 2 hop route (ABC) selected.

OSPF Routers and Link State Advertisements

OSPF is based on a concept of Areas. An Autonomous System (AS) consists of one or more Areas defined by network management. An Area may contain of one or more IP networks.

If an AS does contain more than one area one must be designated as the backbone, area: 0.0.0.0. All Areas (see Router Types) in an AS must be physically connected to the backbone.



Any of the routers shown above could additionally be the Designated Router or Backup Designated Router for its respective network.

Router Types

The location of a router's interfaces with respect to an area determines the type of router it is and the types of Link State Advertisements it exchanges with other routers in that area.

- **Internal Routers** (IR) – A router whose interfaces are within the same area. All Internal Routers compute the shortest path tree to all destinations within its area.
- **Area Border Router** (ABR) – A router with interfaces in different areas but within the same autonomous system. Topological information is gathered (and stored) for each attached area allowing the ABR to compute the shortest path tree for each area separately.
- **Autonomous System Border Router** (ASBR) – A router that acts as a gateway between OSPF and external routes (i.e., routes provided by other routing protocols, static indirect routes, etc.). These routers propagate routes to external networks.
- **Designated Router** (DR) – On broadcast networks (token ring and ethernet) where more than two routers are present only the DR needs to synchronise its link state database with other routers.
- **Backup Designated Router** (BDR) **–** A backup router assumes the responsibilities performed by the DR if that system goes down.

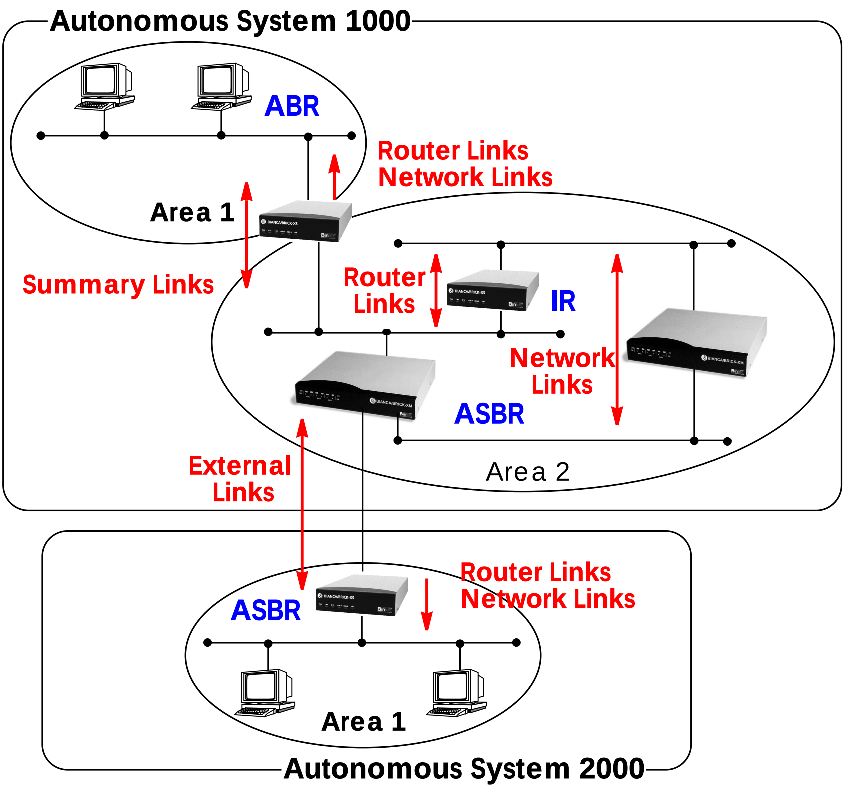
Link State Advertisement Types

OSPF routers exchange routing information via **Link-State Advertisements** (LSAs) that contain information about the networks that can be reached over the router's interfaces.

Link State Advertisements are broken down into five different types shown in the table below.

LSA Type	Purpose:
Router Links	**Generated by**: ALL OSPF Routers **Purpose**: Contains information regarding the state of a router's interfaces within a particular area. Router Links are only flooded within a single area.
Network Links	**Generated by**: The DR (or BDR). **Purpose**: Identifies all OSPF routers present on the network segment and their state. These links are only flooded within a single area.
Summary Links	**Generated by**: Area Border Routers **Purpose**: Identifies the presence of networks within an AS but outside the (local) area. Provides Inter-Area routes allowing routers to learn of networks in other Areas but within the AS.
ASBR Summary Links	**Generated by**: An Area Border Router. **Purpose**: A special type of summary link that provides routes to Autonomous System Border Routers allowing other routers in the AS to find their way out of the system.
External Links	**Generated by**: An Autonomous System Border Router. **Purpose**: Contains information about other Autonomous Systems and allows routers to learn about routes to networks there. External links are flooded into all areas except stub areas.

.



Router Identification

All OSPF routers in an Autonomous System must have a unique Router ID that identifies the router with respect to the AS. Generally an OSPF router's Router ID is taken to be the highest IP address for its first LAN interface.

Initialization

OSPF networks are said to be much “quieter” in comparison to RIP based networks. This is because in OSPF once the initialization phase is complete routing information is only exchanged when link state changes occur. This is much different than with RIP where every 30 seconds a router’s complete routing table is broadcast and verified over the network.

The initialization phase of OSPF is completed once the Link State Database for the area has stabilized and generally occurs once:

1. The OSPF Neighbours have been identified
2. The Designated and Backup Designated Routers have been established.

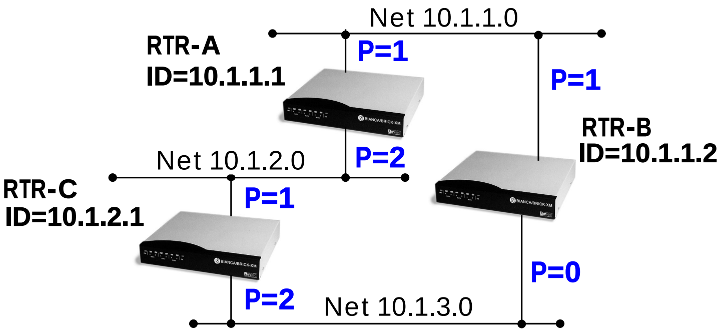
Neighbour Identification and DR/BDR Election

When first coming into service an OSPF router attempts to identify its neighbour OSPF routers using the HELLO protocol. Two router are neighbours if they:

1. Share a common network.
2. Are using the same Area ID for that segment.
3. Are using the same Authentication for the segment.
4. Are using the same parameters (HELLO interval, etc.).

Neighbour routers then decide whether to synchronise their Link State Database (LSDB) with one another. All routers on the segment synchronise their LSDBs with the Designated Router (DR) and the Backup Designated Router (BDR).

When Neighbour routers are identified (via the HELLO protocol) the DR and BDR are also identified. This is sometimes called DR and BDR election and is achieved via IP multicast packets which a router broadcasts via each network segment. For each segment the router with the highest OSPF priority generally becomes the DR. In case of a tie, the router with the higher Router ID becomes the DR.



The DR and BDRs for the three networks shown above would be elected as follows.

Network	DR	BDR
10.1.1.0	RTR-B	RTR-A
10.1.2.0	RTR-A	RTR-C
10.1.3.0	RTR-C	RTR-B

Building the LSDB and the SPT

Link-State Advertisements, contain information about a routers interfaces (i.e.; link's IP address, mask, network type, networks reachable over the link, etc.).

All routers within an area receive all link-state information for all routers in the area. Once synchronized each router has an identical image of the link state database that describes the topological structure of the area.

This database allows each router to separately calculate a **shortest path tree** (SPT), using itself as the root, to any destination in the area. The SPT is used to determine the best interface to route packet. As in RIP the lowest cost route is used however the cost to a destination is calculated differently. In OSPF the cost (or metric) of a link is a function of the bandwidth provided by the link. The higher the bandwidth, the lower the cost.

Authentication

OSPF allows packets containing OSPF routing information to be individually authenticated. Two authentication methods are available which must be configured separately for each network segment.

1. Simple (password) authentication
   A simple text string is sent with each packet. This method is less secure since packet contents can be “sniffed” off the wire using a link analyzer.
2. MD5 (cryptographic) authentication
   When MD5 (Message Digest) is used each packet is appended with a 16 byte encrypted digest. The digest is a function of an authentication key and the contents of the packet. This method is more secure since the key is not sent with the packet.

Note: With MD5 authentication only the digest is encrypted and not the actual contents of the OSPF packet.

OSPF over Demand Circuits

Although OSPF generates less network traffic than RIP, the occasional exchange of routing information (HELLO packets, Link State Database updates or changes, etc.) can lead to increased costs for dial-up interfaces.

To help minimize these costs OSPF on the BRICK has been implemented to include special extensions for Demand Circuits as defined in RFC 1793, *OSPF over Demand Circuits*. These extensions allow for efficient use of dial-up interfaces with OSPF and avoiding excessive ISDN costs. In particular, this means:

1. The exchange of HELLO packets between neighbours is suppressed once the BRICK has synchronized its LSDB with that neighbour (A dial-up connection is initially opened to synchronize the database.).

2. Link State advertisements are only flooded to neighbour routers when an actual change needs to be propagated. Each LSA is marked with a special DoNotAge flag (identifiable by the DC-bit of the LSA or OSPF packet).

Note: ⚠ This feature should only be used if all routers in the AS support this feature (RFC 1793) since some routers don't acknowledge the DC-bit (or use it differently). This could result in unwanted ISDN connections or connections.

Note: ⚠ If a router without RFC 1793 support is removed from the domain in which this feature has been used it is recommended that all OSPF routers be briefly deactivated and reactivated to ensure that all LSAs generated by the removed router are actually flushed.

Enabling Demand Circuit Support

Demand Circuit support for dial-up partner interfaces is enabled by default when an existing interface is enabled for OSPF (AdminStatus is set to active). Support can be manually controlled by setting the interface's ***IfDemand*** object (***ospfIfTable***) to "true" or "false". When set to false, the state of this interface is always up.

Setting this variable to true for one side of the connection is sufficient (that is, as long as OSPF has been enabled on both sides, i.e., ***ipExtIfOspf***=active) if both sides support RFC 1793.

Note: ⚠ Until a neighbour router has been identified HELLO packets are periodically transmitted (default, ***ospfIfPollInterval*** = 120 seconds) over the interface. This results in the link being opened. Once the LSDB has been synchronised, the HELLO protocol is then suppressed.

Import - Export of Routing Information

When different routing protocols are used within the same domain it is sometimes useful to be able to exchange (import or export) routing information between these protocols.

Using the ***ipImportTable*** routing information generated by one protocol (***ipImportSrcProto***) can be imported or exported to another protocol (***ipImportDstProto***).

Currently the following ***SrcProto↔DstProto*** combinations are possible.

		ipImportDstProto	
		rip	ospf
ipImportSrcProto	default_route		✓[1]
	direct		
	static		✓[2]
	rip	–	
	ospf	✓[3]	–

1. ***ipImportSrcProto***=default_route ***ipImportDstProto***=ospf
   This entry forces an external Link State Advertisement to be generated that defines a default route for the Autonomous System.
2. ***ipImportSrcProto***=static ***ipImportDstProto***=ospf
   With this entry statically configured indirect routes will be propagated via OSPF as external LSAs.
3. ***ipImportSrcProto***=ospf ***ipImportDstProto***=rip
   With this entry, all routes learned via OSPF are imported to RIP. If an OSPF route changes the import to RIP will triggered an immediate broadcast of the entire routing table.

The remaining fields of ***ipImportTable*** allow for further control of how (and what) routing information is imported.

- ***ipImportMetric1***
  The metric in the context of the destination protocol the

imported routes should get. If set to -1 these routes get a protocol specific default metric.

- ***ipImportType***
  This object might define protocol specific properties of the imported routes in the context of the destination protocol.
- ***ipImportAddr***
  Specifies (together with ***ipImportMask***) the range of IP addresses for which the table entry should be valid. The entry is valid if the destination IP address of the route lies in the range specified by both objects. If both objects are set to 0.0.0.0, the table entry will be valid for destination.
- ***ipImportMask***
  Together with ***ipImportAddr*** specifies the range of IP addresses for which the table entry should be valid. For example; if Addr=X.X.0.0 and Mask=255.255.0.0 then addresses X.X.0.0 through X.X.255.255 are valid.
- ***ipImportEffect***
  Defines the effect of this entry. If set to “import”, importation from ***SrcProto*** to ***DstProto*** takes place. If set to “doNotImport” importation is prevented.
- ***ipImportIfIndex***
  Specifies the interface index of the interface for which the entry should be valid. If set to 0 (default) the entry is valid for all interfaces.

The OSPF Subsystem

OSPF on the BRICK consists of 11 new system tables and the ospfmon application. An overview of the 10 OSPF tables (ospf group) and the ***ipImportTable*** (ip group) from the SNMP shell:

OSPF System Tables

- ***ospfGeneralGroup***
  Global settings used by the OSPF protocol including the ***ospfAdminStat*** object (must be enabled to use OSPF).
- ***ospfStatTable***
  Status information about Link-State advertisements and OSPF protocol packets that have been sent or received.
- ***ospfErr***
  Status information about bad OSPF packets (bad checksum, incorrect field values, etc.) that have been received.
- ***ospfAreaTable***
  Identifies OSPF areas the BRICK's interfaces are assigned to and logs statistics for each.
- ***ospfLsdbTable***
  Contains header information from the BRICK's Link State Database.
- ***ospfIfTable***
  Lists all OSPF interfaces, their current state, and settings specific to that OSPF interface.
- ***ospfIfMetricTable***
  Lists the actual metric values (or costs) being used for each OSPF interface.
- ***ospfNbrTable***
  Lists the neighbour routers that have been identified via then HELLO protocol and their respective OSPF states.
- ***ospfAreaAggregateTable***
  Specifies IP address ranges for route condensation (also called: inter-area route summarization) among areas.
- ***ospfStubAreaTable***
  Generates a default route for Stub Areas.
- ***ipImportTable***
  Specifies how routes from one routing protocol are imported into another routing protocol.

The ospfmon Application

The ospfmon application can be used from the SNMP shell to display the contents of the BRICK's OSPF Link State Database. Note that only LSA header information is stored in the MIB system tables, this application can be used to dump the complete contents of the database. The various parameters can be used to selectively display specific types of database entries.

Usage:

`ospfmon db` [**`rtr`**|**`net`**|**`sum`** |**`asbr`**|**`ext`**|**`stat`**] <*options*>

Only one of the six identifiers can be used at time to display a cross section of the database.

`rtr`	Show all Router links.
`net`	Show all Network links.
`sum`	Show all Summary links.
`asbr`	Show all AS Border Router links.
`ext`	Show all External Links.
`stat`	Show OSPF database statistics.

Additional options may also be used to further identify more specific types of entries and include:

`area` <id>	Show database entries for area <*id*>.
`rtrid` <*id*>	Show entries generated by router ID <*id*>.
`lsid` <*id*>	Show database entry with link state ID <*id*>.

Example:

Router Links from the Link State Database for Area 0.0.0.0 (from BRICK-XM in the diagram on page) might look like this.

```
BRICK-xm:> ospfmon db rtr area 11.0.0.0
Area 11.0.0.0

    Router Link Age 920     Options 0x20    LsId 192.168.30.1
    RtrId 192.168.30.1   Seq 0x80000002     Checksum 0xe72a Len 48
        options 0x2 links 2
        Stub Network id 12.0.0.2 data 255.255.255.255 metric 1562
        Stub Network id 12.0.0.3 data 255.255.255.255 metric 0

BRICK-xm:>
```

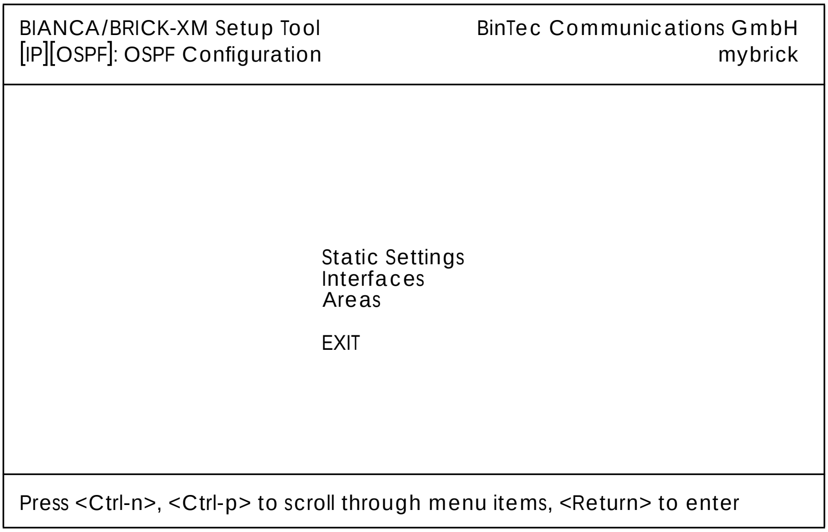
Note that the Link State ID (Lsid) of the database entry has different meanings based on the type of Link State Advertisement that is displayed. The table below shows the meanings for the five LSA types.

LSA Type:	Meaning of Link State ID:
Router Link	The **Router ID** of the router that generated the LSA.
Network Link	The **IP Address** of the DR on the destination network
Summary Link	The ***ipRouteDest*** of the propagated IP route.
ASBR Summary Link	The **Router ID** of the Autonomous System Border Router.
External Link	The ***ipRouteDest*** of the propagated IP route.

OSPF Setup Tool Menus

IP → OSPF →

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



```
BIANCA/BRICK-XM Setup Tool                    BinTec Communications GmbH
[IP][OSPF]: OSPF Configuration                                   mybrick




                              Static Settings
                              Interfaces
                              Areas

                              EXIT




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

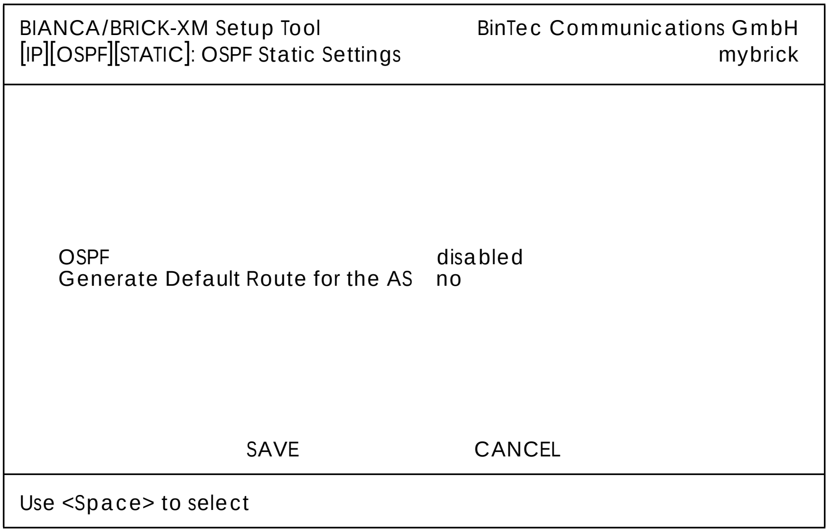
STATIC SETTINGS contains global OSPF parameters. This is where OSPF is enabled on the BRICK.

INTERFACES lists all OSPF capable BRICK interfaces and is used for configuring interface-specific settings.

AREAS lists all known OSPF areas and used for adding/configuring area-specific settings.

IP → **OSPF** → **STATIC SETTINGS**

This menu contains global settings for the OSPF protocol.



```
BIANCA/BRICK-XM Setup Tool                    BinTec Communications GmbH
[IP][OSPF][STATIC]: OSPF Static Settings                          mybrick




    OSPF                                 disabled
    Generate Default Route for the AS    no




                   SAVE                  CANCEL

Use <Space> to select
```

OSPF = Used to enable or disable OSPF. A valid license is also required before OSPF can be used on the BRICK.

Generate Default Route for the AS = When set to yes the BRICK advertises a default route over all active OSPF interfaces (See the “Admin Status” field in the **IP** → **OSPF** → **INTERFACES** menu.).

Note: ⚠ Special consideration should be made when deciding which router is to provide a default route. This router should have the appropriate routes so that it can properly handle traffic for the AS.

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.

IP → **OSPF** → **INTERFACES**

This menu lists the BRICK interfaces OSPF can be configured for. By default, all IP compatible interfaces (present at the time OSPF was enabled) are added to this list and are placed in the passive state.

To configure an interface, scroll to the appropriate entry and hit enter. The fields shown in the resulting EDIT menu shown below can be configured separately for each interface.

```
BIANCA/BRICK-XM Setup Tool                    BinTec Communications GmbH
[IP][OSPF][INTERFACE][EDIT]: Configure Interface en1             mybrick



      Admin Status                   active (propagate routes + run OSPF)
      Area ID                        0.0.0.0

      Metric Determination           auto (ifSpeed)
      Metric (direct routes)         10

      Authentication Type            none
      Authentication Key

      Import indirect static routes  no


                      SAVE                    CANCEL

Use <Space> to select
```

Admin Status = The status of an OSPF interface defines whether routes and / or OSPF protocol packets are propagated over the interface.

If OSPF hasn't been enabled yet only the Admin Status field is displayed (in which case changes are irrelevant).

OSPF routers propagate a Router Link (RL), one per Area, which identifies the router's interfaces in that Area. Both active and passive interfaces are identified in the RL. Status may be active, passive, or off with the following results:

Active	OSPF is running over this interface.
Passive	OSPF is not running over this interface. OSPF protocol packets are neither sent or

	received over the interface, however this interface may be included in other Router Links.
Off	OSPF is not running over this interface and this interface is not included in Router Links.

Note: ⚠ Once an interface is placed in the active state (and saved to memory), OSPF connections may be established over the interface resulting in appropriate costs for dial-up interfaces.

Area ID = Identifies the Area this interface is assigned to.

Metric Determination = Determines how the metric for this interface is calculated. This is the cost of the link that is propagated via link state advertisements.

Determination	Meaning
auto	The metric = the value of the base metric which is based on the bandwidth (***ifSpeed***) of the interface.
fixed	The metric defined (configurable) in the following field is always used (no adjustment).
auto + adjust[a]	When the dial-up interface is in the up state, the metric = <*base metric value*> – 10. Otherwise, metric = <*base metric value*>.
fixed + adjust[a]	When the dial-up interface is in the up state the metric = <*base metric value*> – 10. Otherwise metric = <*base metric value*>.

a. Only valid for Dial-up interfaces.

Metric = Identifies the base metric value, or cost of this interface. For **auto determination** values (see above) the actual metric used is adjusted starting a base metric value which is a simple function of the bandwidth of the physical medium. All interfaces (except leased line interfaces) use the function.

$$\text{Base Metric Value} = \frac{\mathbf{1000{,}000{,}000}}{<\textit{bandwidth in bps}>}$$

This results in 10 for ethernet, 6 for token ring, and 1562 for dialup ISDN interfaces (1 B-Channel). Note that for dialup interfaces the Base Metric Value changes dynamically as ICSDN channels are added/removed while the link is up. For leased line interfaces the base metric is equivalent to the result of the same function less 20 (i.e., 1542 for one leased B-Channel, 781 for two B-channels).

For **fixed determination** values (see previous field) the base metric value can be configured here.

Authentication Type = The type of authentication to use when sending (or verifying incoming) OSPF packets via this OSPF interface. This determines how the key in the Authentication Key field is used.

By default this is set to none. With simple, Key is transmitted as a text string in each packet. With md5, Key is used to create (verify) an encrypted digest which is sent with each packet.

Authentication Key = A text string to use in connection with the Authentication Type set above.

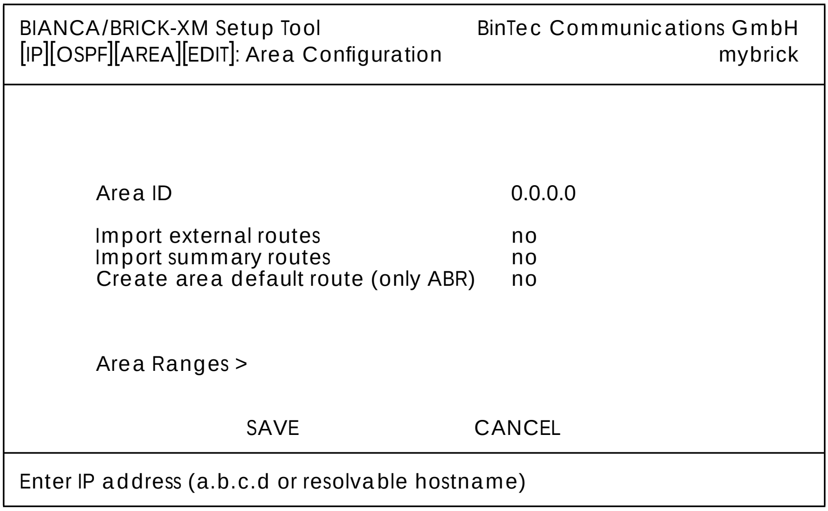
Import indirect static routes = If set to no (default) only direct routes for this interface are propagated over active OSPF interfaces (See the Admin Status field). When set to yes, indirect static routes are also propagated over active interfaces and are contained in external advertisements.

Note: ⚠ Although practical for sites using WAN interfaces without transfer networks caution should be made to avoid routing loops when importing indirect static routes.

IP → OSPF → AREAS

This menu lists the OSPF Areas known to the BRICK. Before an BRICKinterface can be assigned to an Area, the Area ID must first be added here.

The exception is the backbone area which is automatically generated at boot time if no other area is configured and which all interface assignments default to if not explicitly assigned. To edit area-specific settings select the Area ID and hit enter.



```
BIANCA/BRICK-XM Setup Tool                    BinTec Communications GmbH
[IP][OSPF][AREA][EDIT]: Area Configuration                       mybrick



      Area ID                                  0.0.0.0

      Import external routes                   no
      Import summary routes                    no
      Create area default route (only ABR)     no



      Area Ranges >


                 SAVE                  CANCEL

Enter IP address (a.b.c.d or resolvable hostname)
```

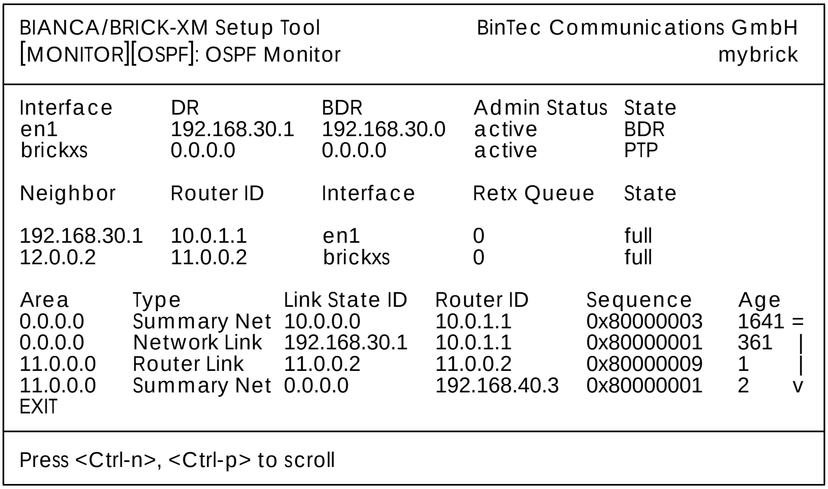
Area ID = Identifies the OSPF Area this entry corresponds to. The backbone area is 0.0.0.0.

Import external routes = Specifies whether external routes should be imported for this area. When set to no, this Area is defined as an OSPF Stub Area.

Area Ranges = This submenu specifies IP Address ranges for route condensation among areas.

MONITORING AND DEBUGGING → OSPF

The OSPF monitor is divided horizontally in three sections and displays information relating to OSPF Interfaces, Neighbours, and Areas.



```
BIANCA/BRICK-XM Setup Tool                  BinTec Communications GmbH
[MONITOR][OSPF]: OSPF Monitor                                  mybrick

Interface     DR            BDR           Admin Status  State
en1           192.168.30.1  192.168.30.0  active        BDR
brickxs       0.0.0.0       0.0.0.0       active        PTP

Neighbor      Router ID     Interface     Retx Queue    State

192.168.30.1  10.0.1.1      en1           0             full
12.0.0.2      11.0.0.2      brickxs       0             full

Area      Type          Link State ID  Router ID     Sequence     Age
0.0.0.0   Summary Net   10.0.0.0       10.0.1.1      0x80000003   1641 =
0.0.0.0   Network Link  192.168.30.1   10.0.1.1      0x80000001   361   |
11.0.0.0  Router Link   11.0.0.2       11.0.0.2      0x80000009   1     |
11.0.0.0  Summary Net   0.0.0.0        192.168.40.3  0x80000001   2     v
EXIT

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

Interfaces Section

The Interfaces section lists all enabled OSPF interfaces (interfaces that have NOT been turned “off” in the IP-OSPF-INTERFACES menu.)

Interface = The BRICK interface the entry corresponds to.

DR = The Designated Router’s IP address on this interface (A DR is not shown for Point-To-Point interfaces).

BDR = The Backup Designated Router’s IP address on this interface (A BDR is not shown for Point-To-Point interface.).

Admin Status = Only active and passive interfaces are shown here (See the IP-OSPF-INTERFACES menu, p. 60).

State = The OSPF status (***ospfIfState***) of the interface shown here may be:

down	OSPF is not running on this interface.
wait	The initial phase of OSPF where DR and BDR are determined.

PTP	The interface is a Point-To-Point interface. No DR or BDR is shown.
DR	The BRICK is the Designated Router for this interface.
BDR	The BRICK is the Backup Designated Router for this interface.
DRother	Another router is the DR/BDR for this interface.

Neighbour Section

The Neighbour section lists the OSPF neighbour routers that have been identified via the HELLO protocol.

Neighbor = The neighbour router's address on this interface.

Router ID = The neighbour router's system wide Router ID.

Interface = The BRICK interface this router was identified over.

Retx Queue = The size of the retransmission queue for this neighbour. This is the number of advertisements that need to be sent to (and acknowledged from) this neighbour.

State = The state of OSPF with this neighbour router may be:

init	The initial phase. A HELLO packet was received from this neighbour.
twoWay	Bidirectional communication with the neighbour. Transmitted HELLO packets have been accepted by the neighbour router (parameters are correct).
EXstart	The exchange of Database Description Packets between the BRICK and neighbour has begun.
exchange	Actively exchanging Database Description Packets with the neighbour router.
loading	The BRICK and the neighbour router are now exchanging Link State Advertisements.
full	The BRICK and neighbour routers' Link State Database are now synchronized.

LSDB Section

The Link State Database section lists the headers for all Link State Advertisements (LSA).

Area = The Area database to which this LSA belongs.

Type = The type of LSA. Five types of LSAs exist: Router Link, Network Link, Summary Link, Summary ASBR, and AS External.

Link State ID = The LSA's Link State ID. The Link State ID's meaning depends on the Type of advertisement.

Router ID = Identifies the router that generated this LSA.

Sequence = This advertisement's sequence number. Sequence numbers allow routers to determine if their database is current or if needs to request an update.

Age = The age (in seconds) of this LSA.

Example OSPF Installation

A typical network installation showing how OSPF could be put to use is shown in the diagram on the following page. Highlights for this setup are as follows:

Area 11.0.0.0 (stub area)

- Since the remote LAN in Area 11.0.0.0 is linked to the backbone via an ISDN dialup link this area is configured as a stub area. This means that external routing information advertisements won't flow into this area. The default route for this area is provided by the router BRICK-xm.
- Because OSPF on the BRICK includes support for Demand Circuits (RFC 1793) the dialup link is only opened when changes in routing information must be propagated.

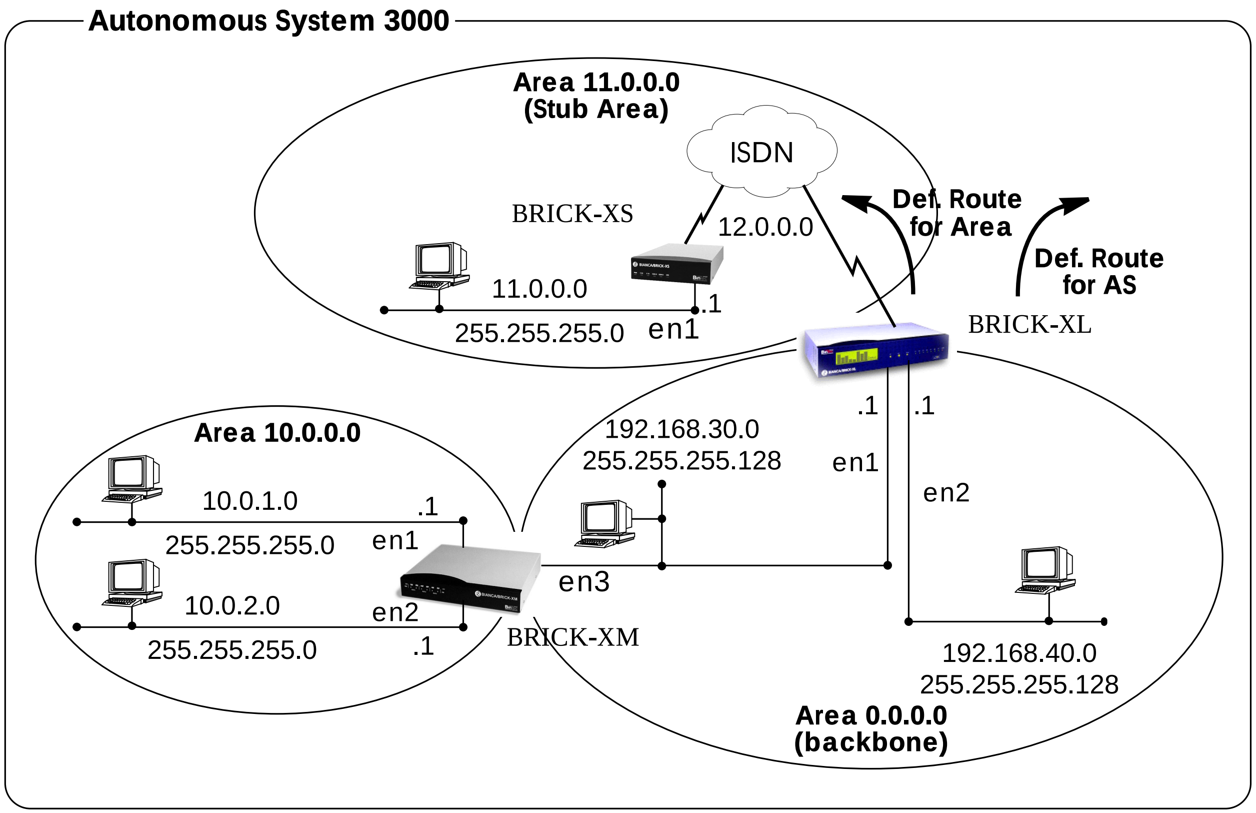
Area 0.0.0.0 (backbone)

- Area 0.0.0.0 is the backbone of the Autonomous System. The router at BRICK-XM will provide the default route for the entire AS and a default route for Area 11.0.0.0.

Area 10.0.0.0

- Area 10.0.0.0 is connected to the backbone via the border router BRICK-XM. Since this is the only link between networks in this area and any external networks (such as the Internet) BRICK-XM will provide Summary Links to routers in other areas. This means that routing information about networks in Area 10.0.0.0 will be combined (or aggregated) into a single advertisement. This lessens the amount of traffic on the backbone and keeps the size of the link state database for area 0.0.0.0 small.

The configuration steps for each BRICK are shown on the following pages.


Autonomous System 3000
Area 11.0.0.0
(Stub Area)
ISDN
BRICK-XS
12.0.0.0
11.0.0.0
.1
255.255.255.0
en1
Def. Route
for Area
Def. Route
for AS
BRICK-XL
.1
.1
en1
en2
Area 10.0.0.0
192.168.30.0
255.255.255.128
10.0.1.0
.1
255.255.255.0
en1
en3
10.0.2.0
en2
255.255.255.0
.1
BRICK-XM
192.168.40.0
255.255.255.128
Area 0.0.0.0
(backbone)

Configuration Overview

All BRICKs:

1. A valid OSPF license must be installed. This can be added to the ***biboAdmLicenseTable*** or from Setup Tool's LICENSES → menu.
2. OSPF must be enabled by setting ***ospfAdminStat*** to **`enabled`**, or from Setup Tool's IP → OSPF → STATIC SETTINGS → menu.

BRICK-XL (overview):

1. Create the dial-up partner interface to BRICK-XS.
2. Have BRICK-XL advertise the default route for the AS.
3. Create the Area entry for Area 11.0.0.0.
4. Assign the new dialup partner interface to Area 11.0.0.0 and set the interface to active.
5. Verify ethernet interfaces en1 and en2 are assigned to Area 0.0.0.0 and set both interfaces to active.

BRICK-XS (overview):

1. Create the dial-up partner interface to BRICK-XL.
2. Create the Area entry for Area 11.0.0.0.
3. Assign the ethernet interface (en1) to Area 11.0.0.0 and set the interface to active.
4. Assign the new dial-up interface to Area 0.0.0.0 and set the interface to active.

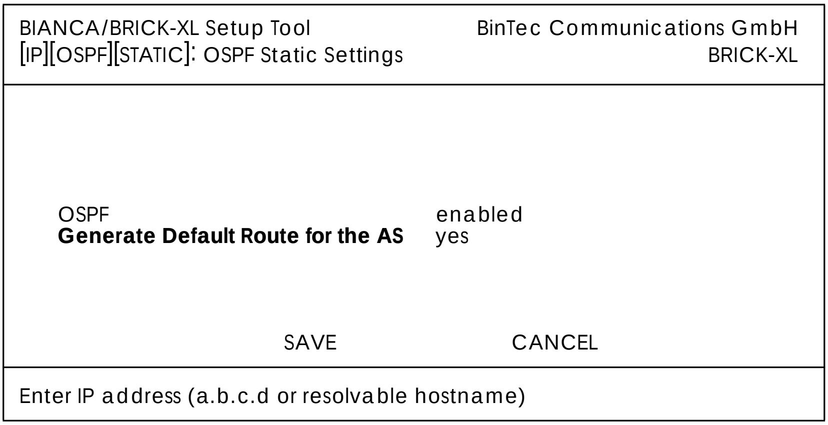
BRICK-XM (overview):

1. Create the Area entry for Area 10.0.0.0.
2. Assign ethernet interfaces en1 and en2 to Area 10.0.0.0 and set both interfaces to active.
3. Verify ethernet interface en3 is assigned to Area 11.0.0.0 and set the interface to active.
4. Create the OSPF aggregate for the LANs attached to en1 and en2 to reduce the routing traffic sent over en3.

Configuration Steps for BRICK-XL

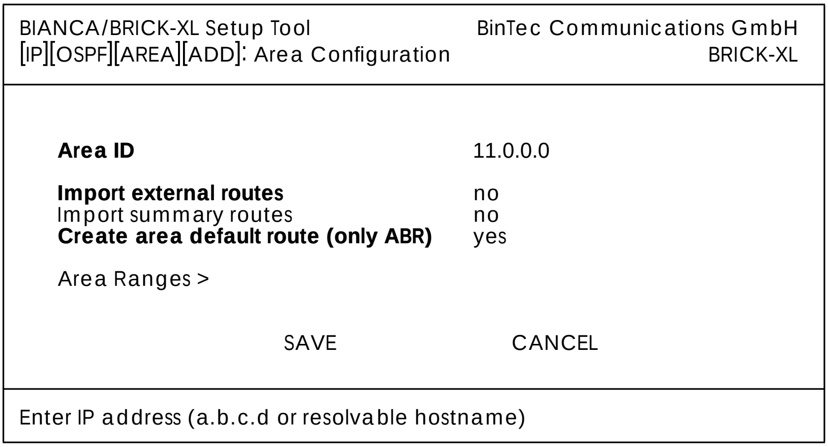
Step 1 Assuming an OSPF license is installed and OSPF has been enabled the partner interface to BRICK-XS should be created. Note that our example uses a transfer network (network 12.0.0.0).

Step 2 Since BRICK-XL should advertise the default route for the AS set this field to yes in IP → OSPF → STATIC SETTINGS →.



```
BIANCA/BRICK-XL Setup Tool                BinTec Communications GmbH
[IP][OSPF][STATIC]: OSPF Static Settings                    BRICK-XL

    OSPF                                 enabled
    Generate Default Route for the AS    yes

                      SAVE                    CANCEL

Enter IP address (a.b.c.d or resolvable hostname)
```

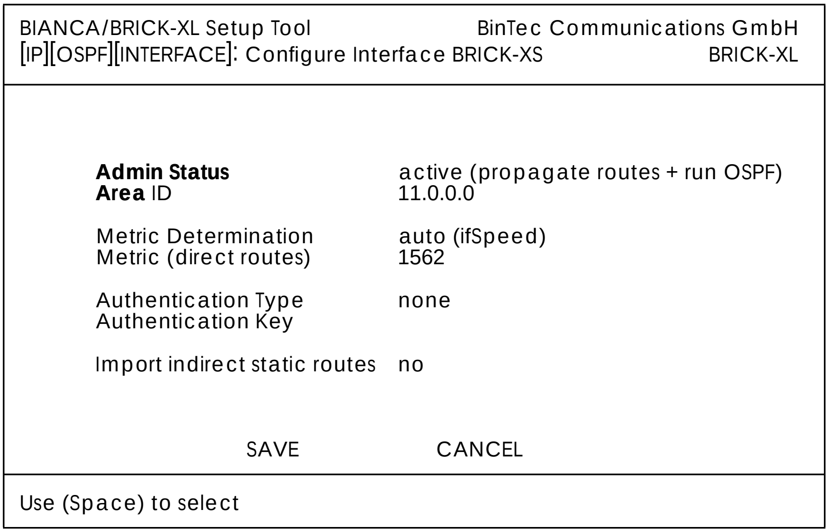
Step 3 In the IP → OSPF → AREAS → menu create an entry for Area 11.0.0.0. Define this area as a Stub Area and have BRICK-XL generate the default route for this area.



```
BIANCA/BRICK-XL Setup Tool                BinTec Communications GmbH
[IP][OSPF][AREA][ADD]: Area Configuration                   BRICK-XL

    Area ID                                  11.0.0.0

    Import external routes                   no
    Import summary routes                    no
    Create area default route (only ABR)     yes

    Area Ranges >

                      SAVE                    CANCEL

Enter IP address (a.b.c.d or resolvable hostname)
```

Step 4 In the IP → OSPF → INTERFACES → menu locate the dialup interface entry created in step 1 and hit enter to edit the settings.

Set the Admin Status to active and assign it to Area 11.0.0.0 (or the area created in step 3) and select SAVE .



```
BIANCA/BRICK-XL Setup Tool                    BinTec Communications GmbH
[IP][OSPF][INTERFACE]: Configure Interface BRICK-XS             BRICK-XL



     Admin Status                active (propagate routes + run OSPF)
     Area ID                     11.0.0.0

     Metric Determination        auto (ifSpeed)
     Metric (direct routes)      1562

     Authentication Type         none
     Authentication Key

     Import indirect static routes  no


                    SAVE              CANCEL

Use (Space) to select
```

By default, dial-up interfaces are set to passive in the Admin Status field.

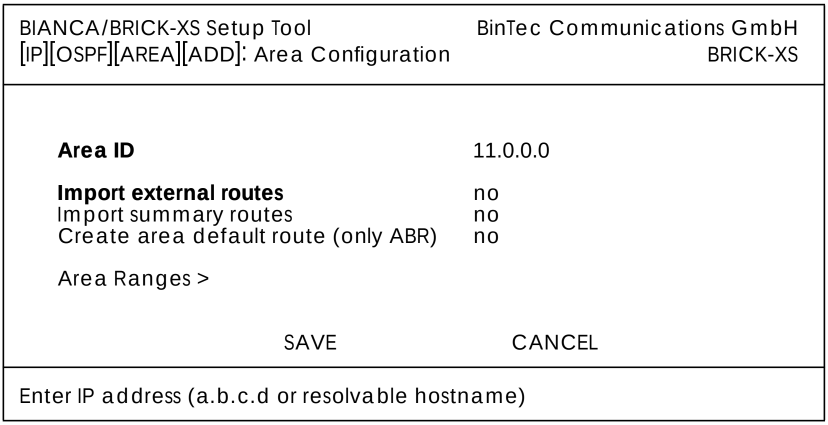
Step 5 In IP → OSPF → INTERFACES → menu verify the ethernet interfaces en1 and en2 are assigned to the backbone, (Area 0.0.0.0 which is the default area).

Set the Admin Status to active and assign it to Area 11.0.0.0 (or the value from step 2) and select SAVE

Configuration Steps for BRICK-XS

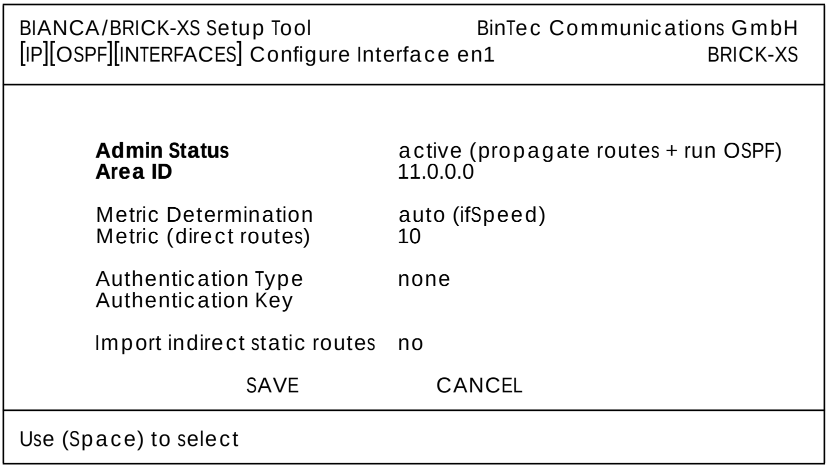
Step 1 Assuming an OSPF license is installed and OSPF has been enabled the dial-up partner interface to BRICK-XL should be created. In our example a transfer network (12.0.0.0) is used.

Step 2 In the IP → OSPF → AREAS → menu create Area 11.0.0.0. and define it as a Stub Area.



```
BIANCA/BRICK-XS Setup Tool                BinTec Communications GmbH
[IP][OSPF][AREA][ADD]: Area Configuration                   BRICK-XS


   Area ID                                 11.0.0.0

   Import external routes                  no
   Import summary routes                   no
   Create area default route (only ABR)    no

   Area Ranges >


                    SAVE                  CANCEL

Enter IP address (a.b.c.d or resolvable hostname)
```

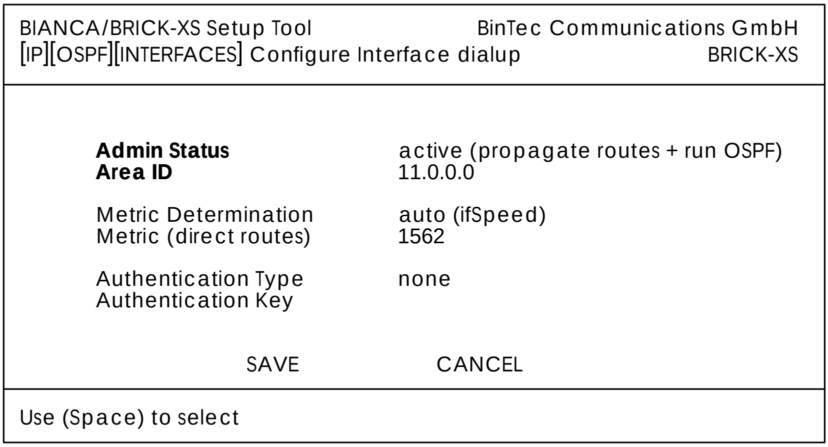
Step 3 In the IP → OSPF → INTERFACES → menu assign the ethernet interface (en1) to Area 11.0.0.0 and make sure the Admin Status is set to active.



```
BIANCA/BRICK-XS Setup Tool                BinTec Communications GmbH
[IP][OSPF][INTERFACES] Configure Interface en1              BRICK-XS


      Admin Status                 active (propagate routes + run OSPF)
      Area ID                      11.0.0.0

      Metric Determination         auto (ifSpeed)
      Metric (direct routes)       10

      Authentication Type          none
      Authentication Key

      Import indirect static routes  no

                  SAVE            CANCEL

Use (Space) to select
```

Step 4 In IP → OSPF → INTERFACES → menu locate the dialup interface (created in step 1) and assign the interface to Area 11.0.0.0 (or the value used in step 2).

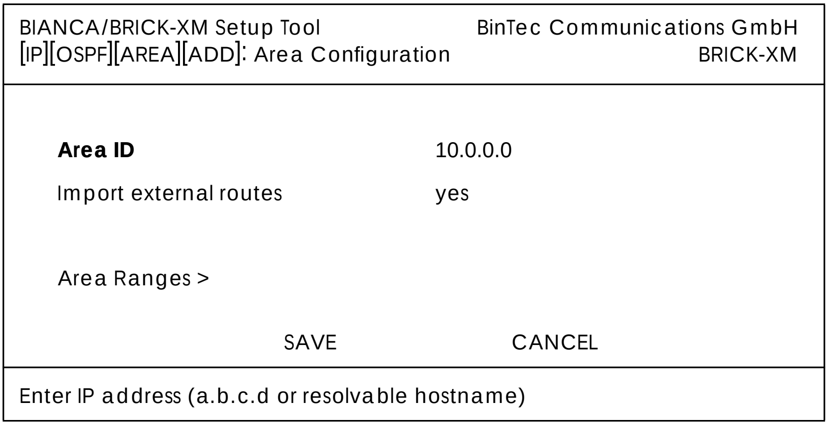
Set the Admin Status for the dialup interface to active and select SAVE.



```
BIANCA/BRICK-XS Setup Tool                    BinTec Communications GmbH
[IP][OSPF][INTERFACES] Configure Interface dialup               BRICK-XS

          Admin Status              active (propagate routes + run OSPF)
          Area ID                   11.0.0.0

          Metric Determination      auto (ifSpeed)
          Metric (direct routes)    1562

          Authentication Type       none
          Authentication Key

                      SAVE                CANCEL

Use (Space) to select
```

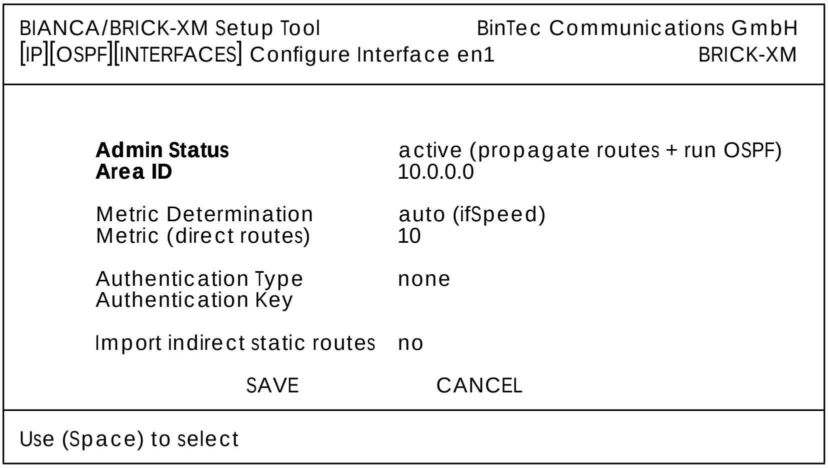
Configuration Steps for BRICK-XM

Step 1 An OSPF license must already be installed and OSPF should been enabled **IP** → **OSPF** → **STATIC SETTINGS** → menu. Then create an area entry for Area 10.0.0.0 in the **IP** → **OSPF** → **AREAS** → menu.



```
BIANCA/BRICK-XM Setup Tool                    BinTec Communications GmbH
[IP][OSPF][AREA][ADD]: Area Configuration                       BRICK-XM


   Area ID                              10.0.0.0

   Import external routes               yes


   Area Ranges >

                    SAVE                    CANCEL

Enter IP address (a.b.c.d or resolvable hostname)
```

Step 2 In the **IP** → **OSPF** → **INTERFACES** → menu assign ethernet interfaces en1 and en2 to Area 10.0.0.0 (or the value from the previous step) and set the Admin Status for each interface to active.



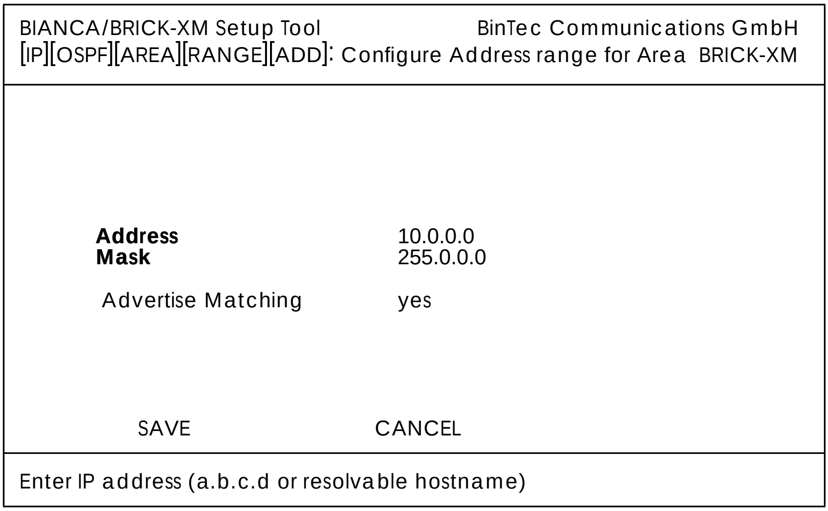
```
BIANCA/BRICK-XM Setup Tool                    BinTec Communications GmbH
[IP][OSPF][INTERFACES] Configure Interface en1                  BRICK-XM


      Admin Status                 active (propagate routes + run OSPF)
      Area ID                      10.0.0.0

      Metric Determination         auto (ifSpeed)
      Metric (direct routes)       10

      Authentication Type          none
      Authentication Key

      Import indirect static routes  no

                  SAVE                 CANCEL

Use (Space) to select
```

Step 3 Ethernet interface en3 should already be assigned to the backbone, Area 0.0.0.0 which is the default.

In the IP → OSPF → INTERFACES → menu verify this setting and change the Admin Status to active.

Step 4 Return to the IP → OSPF → AREA → menu and scroll to the Area ID entry for the backbone and hit enter.

Move to the AREA RANGES → submenu to add an OSPF aggregate for the LANs attached to en1 and en2. The Address and Mask entries shown below will match any routes with a destinations starting with 10, or 10.*.*.*.



```
BIANCA/BRICK-XM Setup Tool                    BinTec Communications GmbH
[IP][OSPF][AREA][RANGE][ADD]: Configure Address range for Area  BRICK-XM




      Address                          10.0.0.0
      Mask                             255.0.0.0
      Advertise Matching               yes



         SAVE                    CANCEL

Enter IP address (a.b.c.d or resolvable hostname)
```

This entry means that BRICK-XM will consolidate multiple routes (routes for destinations in Area 10.0.0.0) into a single link state advertisement.

This will effectively reduce the amount of traffic sent over the backbone as will help keep the size of the link state database and routing tables for routers in other areas to a minimum.

Troubleshooting OSPF

When troubleshooting OSPF you should check the following things first.

- **Is a valid OSPF license installed**?
  Check the ***biboAdmLicInfoTable***.
- **Is OSPF enabled?**
  Set ***ospfAdminStat*** to “enable”.
- **Have all OSPF Areas been configured**?
  Check the ***ospfAreaTable***.
- **Are all OSPF interfaces assigned to the desired areas**?
  Check the interface’s ***IfAreaId*** in the ***ospfIfTable***.
- **Is the Admin Status of each interfaces configured properly**?
  Check the value of ***ipExtIfOspf*** for the interface.
- **Have all OSPF neighbour routers been identified**?
  OSPF neighbour routers identified via the HELLO protocol should appear in the ***ospfNbrTable***.
- **If other OSPF routers are present on the network but haven’t been identified.**
  - Verify the interface parameters are the same for all routers in the area. Check: ***ipRouteMask***, ***ospfIfAreaID***, ***ospfIfHelloInterval***, ***ospfIfRtrDeadInterval***, ***ospfIfAuthKey***, ***ospfIfAuthType***).
  - Verify the area parameters are the same for all routers in the area. Check: ***ospfImportAsExtern***.
- **Has the DR and BDR been elected for broadcast nets?**
  Check the addresses set in the ***ospfIfDesignatedRouter*** and ***ospfIfBackupDesignatedRouter*** objects.
- **Are OSPF syslogs appearing in *biboAdmSyslogTable***?
  First set ***biboAdmSyslogTableLevel*** to “debug”.
- **Is NAT turned off for all OSPF interfaces?**
  Check the ***Nat*** field in ***ipExtIfTable***. It must be “off”.

Other error conditions may be determined by viewing the contents of the ***ospfErrTable*** and ***ospfStatTable***.