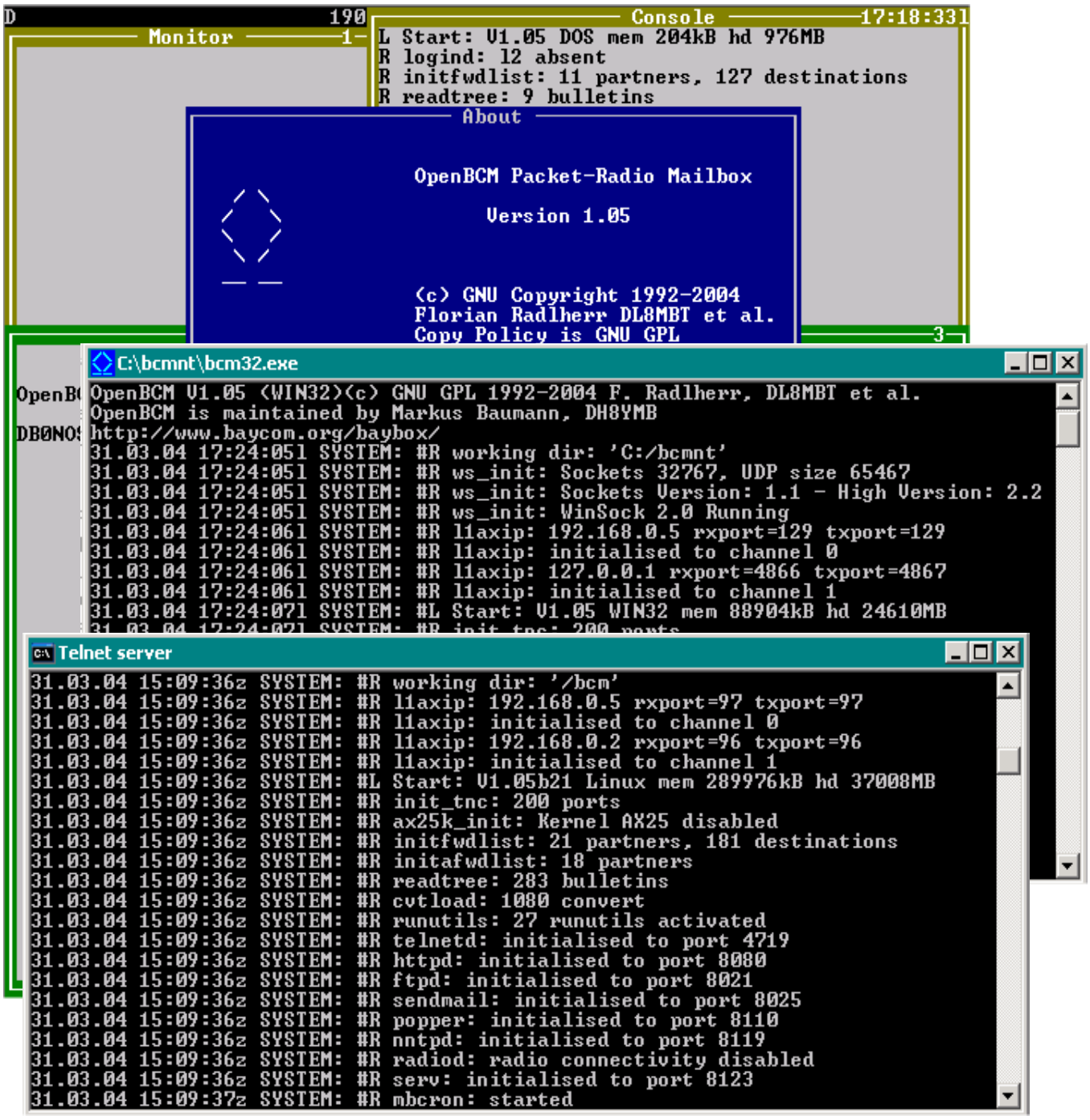
Documentation
for OpenBCM v1.07 Beta

The complete documentation for installation and usage of an OpenBCM mailbox system



Original documentation by Florian Radlherr DL8MBT <flori(at)baycom.org>,
additional texts by
Dietmar Zlabinger OE3DZW <dietmar(at)baycom.org>,
Johann Hanne DH3MB <jonny(at)baycom.org> and
Markus Baumann DH8YMB <dh8ymb(at)web.de>

Last edited: 29. Dezember 2006

Table of contents

NOTE:
All texts in green are not yet translated from german into english language!

1. Introduction and general notes

1.1. Introduction

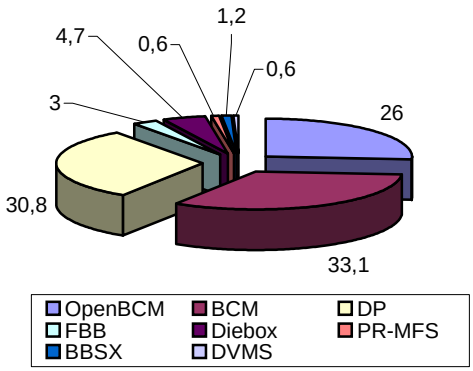
Zwei Jahre sind nun seit dem letzten Release von OpenBCM vergangen. In der Zwischenzeit sind rund 60 Betaversionen bei den verschiedensten Mailboxen und in den unterschiedlichsten Konfigurationen ausgetestet worden.

Pünktlich zur 22. PR-Tagung in Darmstadt erscheint nun OpenBCM v1.06. Als größte Neuerung gilt hier sicherlich die Implementation des ACTIVE ROUTING.

Im Vorwort zur v1.05 wurden die damals (März 2004) aktuellen Nutzungsdaten von Mailboxsoftware im deutschen Packet Radio Netzwerk vorgestellt:

Mailboxanzahl und prozentualer Anteil:

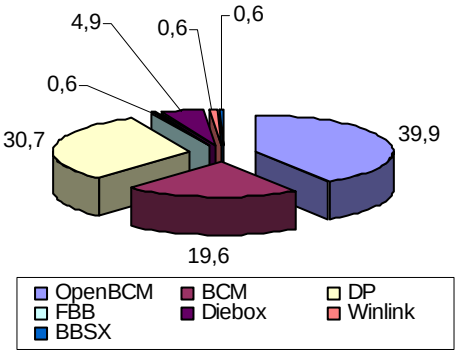
44	OpenBCM	26,0%
56	BCM	33,1%
52	DP	30,8%
2	BBSX	1,2%
1	DVMS	0,6%
1	PR-MFS	0,6%
5	FBB	3,0%
8	DieBox	4,7%
169	Boxen in DL	



Nach 2 Jahren lohnt nun wieder einmal der Blick über die Mailbox-Landschaft. Erfreulich für das OpenBCM-Projekt ist der starke Zuwachs an OpenBCM-Nutzung – von 26% anno 2004 auf jetzt knapp 40%.

Mailboxanzahl und prozentualer Anteil:

65	OpenBCM	39,9%
32	BCM	19,6%
50	DP	30,7%
1	BBSX	0,6%
1	FBB	0,6%
1	Winlink	0,6%
8	DieBox	4,9%
163	Boxen in DL	



Denjenigen Sysops, die noch eine alte BCM/OpenBCM-Version einsetzen, sei nochmals dringend empfohlen, auf die aktuelle OpenBCM v1.06 umzusteigen. Was insbesondere bei einem Update von BCM v1.42n auf OpenBCM v1.06 beachtet werden sollte, ist im Kapitel "23. Upgrade from BCM v1.42n to OpenBCM" zusammengefasst.

Es bleibt am Ende des Vorwortes wieder einmal übrig, noch denen zu danken, die durch endlose Tests und qualifizierte Berichte zur heute verfügbaren Stabilität und Funktionalität beigetragen haben, sowie all denen, die sich aktiv an Weiterentwicklung beteiligt haben!

31.03.2006 - Markus, DH8YMB

1.2. Historical overview

- August 1991: Erster Entwurf einer Spezifikation für ein Mailbox-Dateisystem. Diese war der heutigen Realisierung bereits sehr ähnlich.
- Januar 1992: Entwurf und Entwicklung einer universellen Benutzeroberfläche mit Editor und mit Multitasking-Kern. Bei der Entstehung dieser Teile wurde noch nicht an einen Einsatz in einem Mailboxprogramm gedacht.
- Mai 1992: Entscheidung: "Es wird eine Boxsoftware entwickelt"; Entwicklung von ersten Modulen für die Mailbox, insbesondere BID-Verwaltung, Userdaten-Verwaltung, Suchfunktionen, Datei-Semaphoren, etc.
- Juni 1992: Erstellung des Boxgerüstes, TNC-Anschluss, Kommandoauswertung, etc.
- Juli 1992: Integration der Mailboxteile mit der Benutzeroberfläche, Einsatz des Multitasking-Schedulers
- August 1992: Erstellung und Test der Store&Forward-Routinen, erste Systemtests über Funk unter dem Rufzeichen DF0AR
- September 1992: Inbetriebnahme der neuen Box bei DB0AAB. Zunächst am Standort von DL8MBT über einen Funklink. Nach fortgeschrittener Fehlerbehebung Inbetriebnahme bei DB0AAB auf der Fachhochschule München
- Oktober 1992: Inbetriebnahme der BayCom-Mailbox an den Knoten DB0LNA und DB0RGB
- November 1992: Herausgabe der Version 1.00 auf der Interradio in Hannover
- Februar 1994: Version 1.17, Mehrsprachigkeit der Mailbox
- September 1994: Binäre Nachrichten
- November 1994: Version 1.35, erste Linux-Version
- Februar 1995: Version 1.36
- Juni 1995: Vorstellung von PC/FlexNet auf der HamRadio, erste FlexNet-Version der Mailbox
- Januar 1996: Version 1.37 für FlexNet 3.3e
- Oktober 1996: Version 1.38 für Linux und Windows NT, User-S&F
- Juni 1997: Version 1.39 mit HTTP-Interface
- Juni 1998: Version 1.40 mit einer Fülle an neuen Features: 7plus-Erkennung, komprimierter S&F nach F6FBB, Mailing-Listserver, File-Server, MD2/MD5-Passwortverfahren, YAPP-Protokoll, neues Run-Utility-Interface, Unterstützung des Linux Kernel-AX.25 und einer Menge Bugfixes
- November 1999: Version 1.42 Jahr-2000-fähig, neues Protokoll für die Weiterleitung von White-Page-Informationen, Bugfixes, deutlich verbesserte Unterstützung von CB-Funk. Neue Lizenz (GPL).
- Oktober 2001: Version 1.44 zahlreiche neue Features, viele kleinere Änderungen am Quellcode, u. a. diverse Patche von DF3VI
- September 2002: Kurz nach dem Erscheinen der Version 1.46 wird der Name von "BCM" in "OpenBCM" geändert. Der Versionzähler beginnt mit dem neuen Namen nun wieder bei 1.00.
- Dezember 2003: OpenBCM v1.04 wird freigegeben, u. a. mit Patchen von DB1RAS
- April 2004: OpenBCM v1.05, enthält u. a. FTP-Server von DH6BB
- März 2006: OpenBCM v1.06 nach rund 60 Betaversionen veröffentlicht

Die komplette Versionshistorie ist den gesondert erhältlichen Dokumenten *history.txt* (hier sind alle Änderungen von BCM v1.00 bis v1.46 zusammengefasst) bzw. *changes.txt* (alle Änderungen ab OpenBCM v1.00) zu entnehmen.

1.3. Switching from BCM to OpenBCM

The project "Baycom-Mailbox" was already looked after from a huge number of people, who worked on the source sometimes more, sometimes less intensive. This is not amazing, because the radio hobby should be a hobby and not a fulltime job.

Thus the main concept of this mailbox software was developed by Flori DL8MBT. Starting from BCM v1.36 above especially Dietmar OE3DZW worked hard on the sourcecode. However, code from Jonny DH3MB, Deti DG9MHZ and a lot of other people were included.

In the spring 2000 the Baybox project has been taken over by Markus DH8YMB. From this time a re-orientation phase began. Most notably Wolfgang DK2UI was the leading person in this time, who works on the sources and includes new ideas.

In September 2002, with BCM v1.46, a final stroke was drawn under this phase. The project was renamed into OpenBCM and is maintained only by Markus DH8YMB from this time. The new name should not symbolise only a restart but the prefix "open" should underline the openness of the project.

Everybody, who wants to add something useful to the project is welcome! Send ideas and source code to include in future versions.

1.4. Notes to this documentation

This documentation will often refer to files. The filenames are always written in italic, so that it is easy to see, that a filename is refered.

Note: When using Linux you have to distinguish between filenames *rhosts.bcm*, *RHOSTS.BCM* and *Rhosts.bcm*; when using DOS/Windows it is unimportant. Nevertheless, it is recommended to use always filenames in lower case. This is also the reason why all filenames of OpenBCM will appear in this docu always in lower case.

1.5. Version numbers of OpenBCM

In the future there will be not so much release versions of OpenBCM like in the past of BCM. New features will be implemented and tested in so called beta versions. These beta versions can be downloaded and tested by everybody, but you have always keep in mind that some problems and failures may happen when using these versions. You can see the difference, if a version is a beta or release version in the version number. "x.xxbx" means beta version: b = Beta, while the x symbolize a number. A release version of OpenBCM will use the syntax "x.xx".

2. Functions and maximum ratings

2.1. Main features

- forwarding like W0RLI, including additions of DF3AV (support of lifetime, system messages for board E (Erase) and M (MyBBS), 8-character-long board names)
- forwarding like F6FBB, including checksum saved compression and resume plus additions of DL8HBS
- support and analysis of hierarchical forward addresses
- support of file transfer protocols AutoBIN, YAPP and Didadit
- user basic commands: DIR, LIST, READ, SEND, ERASE, REPLY, ALTER, QUIT, HELP, CHECK, MYBBS
- usual command syntax like "read aktuell 1-3" or "send network @ ww New digi"

- hierarchical board structure
- WPROT support

2.2. Difference to other mailbox systems anno 1992

The user interface of OpenBCM is similar to OE5DXL mailbox system. Nevertheless the compatibility to other mailbox systems has been not ignored, in first line DieBox/TheBox from DF3AV. However there are some specials:

- The command DIR doesn't show (like DieBox and DPBox) a list of all available boards. It works similar to command LIST, but has some more sub commands like DIR BOARDS or DIR USERS.

- When executing command DIR (or LIST) all mails are listed in the order the have been arrived in the mailbox (not like the order of OE5DXL). To list the newest mails the area specification is a little bit different. Examples:

```
d 1-5       lists the first (means oldest) 5 mails
d -5        lists the last (means newest) 5 mails
```

2.3. Maximum ratings

Beim Design der OpenBCM-Mailbox wurde darauf geachtet, dass möglichst wenige Betriebsparameter durch interne Anschläge begrenzt werden. Überall, wo jedoch Speicherplatz nicht sinnvoll dynamisch belegt werden konnte, sondern statisch reserviert werden muss, oder wo Wertebereiche von Variablen an ihren Grenzen sind, treten Maximalwerte auf. Diese können mit dem Befehl "status limits" in der Mailbox abgefragt werden. Wegen Speichermangels sind die Limits unter DOS knapp gesetzt, unter Linux/Windows können diese Limits im Allgemeinen leicht erhöht werden.

3. Selection of PC hardware and operating system

3.1. Which PC?

Die Geschwindigkeit der Mailbox wird hauptsächlich durch die Festplatte, deren Controller und das verwendete Dateisystem bestimmt. Bei einem 486 DX mit 50 MHz und einer IDE-Festplatte werden unter DOS etwa 100 kB/s erreicht. Bei einem modernen, schnellen Rechner liegen unter Linux mit Extended2-Dateisystem die Werte entsprechend höher. Ein wenig spielt nur auch die Leistung der CPU eine Rolle.

Generell gilt: je schneller der PC, umso besser. Es muss aber für einen Mailbox-PC, der an einem automatischen Digipeater-Standort rumsteht, nicht der schnellste, neuste und teuerste PC einkalkuliert werden, ein guter Gebrauchter (z.B. Pentium3-Klasse) reicht heutzutage auch unter Linux oder Windows völlig aus.

Sparen kann man bei einer Mailbox außerdem bei der Grafikkarte und dem Monitor. Wer dafür eine 3D-Karte und einen 19-Zoll-Monitor verwendet, ist selbst schuld.

3.2. Needed harddisc space

Neben der Rechner-Performance ist auch die Kapazität des Speichermediums für eine Mailbox wichtig.

Der erforderliche Plattenplatz für OpenBCM beträgt mindestens 80 MB, wenn nur User-Mails gespeichert werden sollen. Die System-Dateien (User-Daten, BID-Verwaltung, etc.) sind nicht auf Plattenersparnis ausgelegt, es sollte unbedingt ein Platz von etwa 50 MB für diese Dinge reserviert werden. Bei umfangreichen Archivboxen werden, je nach Menge an Mails und Zusatzdateien (z.B. für den Filesurf), sicherlich auch höhere Werte von bis zu mehreren Gigabyte verlangt.

All diese Werte sollten heutzutage jedoch bei den Lesern ein Grinsen auf die Lippen werfen; werden doch momentan Festplatten in den Größen von mehreren Dutzend Gigabyte verkauft. Eine solche moderne Festplatte ist also mit Sicherheit ausreichend!

Gedanken kann man sich eher machen, wie man die Datenmengen vernünftig sichert. So ist es sinnvoll, z.B. ein RAID 1 Array (Spiegelung) aus mehreren Festplatten in den Mailboxrechner einzubauen. Fällt hier eine Festplatte aufgrund eines Hardwaredefektes aus, kommt es zu keinem Datenverlust!

Als grobe Faustregel zur Festplattengröße kann folgende Tabelle betrachtet werden:

Boxgröße	Usermails	Bulletins	Lifetime User (Tage)	Lifetime Bulletins (Tage)	HDD System (MB)	HDD User (MB)	HDD Bulletins (MB)	HDD Gesamt (MB)
sehr klein	ja	nein	120	--	60	20	--	80
klein	ja	ja	360	60	70	50	400	540
mittel	ja	ja	360	180	70	100	500	700
groß	ja	ja	∞	∞	80	>500	>8000	>10000

Soll ein Digipeater auf dem gleichen Rechner mitlaufen, wird auch hierfür Speicherplatz und CPU-Leistung vom PC benötigt. Außerdem wird empfohlen, immer mindestens 500 MB auf der Platte frei zu lassen, um Backups und eine Vergrößerung der Verwaltungsinformationen jederzeit zuzulassen.

3.3. Which operating system is the best?

1991 begann die Entwicklung der BayCom-Mailbox Software. Die damalige Wahl des Betriebssystems DOS geschah nach folgenden Gesichtspunkten:

- Vorhandensein einer passenden Entwicklungsumgebung (vom BayCom-Terminal und -Node)
- große Verbreitung und günstiger Preis, daher Akzeptanz zum Einsatz bei bereits laufenden bzw. neu zu installierenden Systemen
- direkter Hardwarezugriff, dadurch einfache Möglichkeiten für PR-Anbindung

Bereits zum Zeitpunkt der Entwicklung waren natürlich auch die Schwächen von DOS schon bekannt. Diese sind in erster Linie:

- Kleiner direkt adressierbarer Speicher unter 1 MB, dadurch permanente Speicherknappheit
- Segmentierung mit max. 64 kB langen linearen Speicherblöcken
- theoretisch maximal ca. 63000 Dateien pro Partition auf der Platte, praktisch noch weniger
- große Verschwendung durch das FAT-Dateisystem bei großen Platten
- schlechte Wartbarkeit, weil nur jeweils ein Programm laufen kann (kein Multitasking)

- schlechte Systemstabilität, bei Programmabsturz hängt i.A. der gesamte Rechner

Es war also im Prinzip bereits am Anfang bekannt, dass DOS nicht die ideale Wahl für den Betrieb einer Mailbox ist. Trotzdem wird es sich noch lange halten, weil das Gesetz der Trägheit und die Tatsache, dass jede Änderung mit Kosten verbunden ist, dies erzwingen.

Interessant ist die Tatsache, dass es kein anderes Betriebssystem mit ähnlich schlechten Eigenschaften wie DOS gibt. Das mag wohl daran liegen, dass DOS das erste und älteste PC-Betriebssystem ist.

Folgende Alternativen stehen potentiell zur Auswahl:

Windows 95/98/ME:
(+) Installation/Preis: einfach + bezahlbar
(-) Dateisystem: wie DOS, folglich kein Vorteil, evtl. FAT32
(+) großer Verbreitungsgrad, somit viele verfügbare Entwickler
(-) mittlerweile wird nur noch Windows ME von Microsoft supported

OS/2:
(-) Installation/Preis: einfach + bezahlbar
(o) Dateisystem: HPFS, besser als DOS-FAT
(-) hohe Hardwareanforderungen im Vergleich zu DOS durch graphische Oberfläche, besser Systemstabilität als DOS
(-) wird von IBM nicht mehr supported

Windows NT/2000/XP:
(+) Installation/Preis: einfach, aber recht teuer
(o) Dateisystem: NTFS, weit besser als DOS-FAT
(-) hohe Hardwareanforderungen im Vergleich zu DOS durch graphische Oberfläche, aber hohe Systemstabilität

Linux:
(+) Installation: je nach Distribution von einfach bis umständlich und fehlerträchtig, dafür aber kostenlos
(+) Dateisystem: ext2-Dateisystem sehr schnell und effizient
(+) graphische Bedienoberfläche nicht erforderlich, daher vergleichbar mit DOS, System offen

Andere UNIX-Derivate für PC wurden nicht betrachtet, da der Preis meist zu hoch oder die Beschaffung schwierig ist.

Unter dem Strich fallen die Betriebsysteme OS/2 und Windows95/98/ME schon auf Grund des nicht mehr gewährleistetem Support aus.

Als DOS-Alternativen bleiben somit Windows NT/2000/XP und Linux über. Es soll hier nicht zur Diskussion angeregt werden, welches der genannten Systeme nun das Beste ist und welches nicht, trotzdem zeigt sich auf Grund der Vor-/Nachteile, das das Linux-System wohl als am ehesten geeignet ist, den Ansprüchen eines Mailbox-Systems gerecht zu werden.

3.3.1. Linux

3.3.1.1. Why Linux?

Wie auch immer die Vor- und Nachteile der einzelnen Systeme ausfallen mögen, die erste Portierung der OpenBCM-Mailbox wurde für Linux durchgeführt.

Die ersten Schritte dieser Portierung haben schon Mitte 1994 stattgefunden. Es war weniger die unbedingte Notwendigkeit, als vielmehr der technische Reiz, der zu diesem Schritt geführt hat. So ist es auch nicht verwunderlich, dass nach den ersten Erfolgserlebnissen dieser Zweck erfüllt war, und das Projekt wieder in die Ecke geworfen wurde.

Einige Leute haben sich mit der damaligen BCM Version 1.35a "vergnügt". Diese war allerdings für den praktischen Einsatz vollkommen unbrauchbar und vermochte lediglich das Erlebnis "es geht prinzipiell" zu demonstrieren.

Im Dezember 1995 brach dann erneutes Interesse aus und das Projekt wurde aus der tiefen Versenkung ausgegraben. Da der Code größtenteils portabel ist, konnte auf die zwischenzeitliche Weiterentwicklung der DOS-Version problemlos aufgesetzt werden.

Komplexere Betriebssysteme haben zumeist die Eigenschaft, auf einer großzügig bemessenen Hardware sehr viel bessere Leistungseigenschaften zu besitzen als z.B. DOS. Das betrifft sowohl die Dateizugriffe als auch den Umgang mit Arbeitsspeicher. Ist der Rechner jedoch unangemessen klein, so brauchen die Mechanismen innerhalb des Betriebssystems einen großen Anteil der zur Verfügung stehenden Resourcen und bewirken daher keine Leistungssteigerung, sondern eher ein Ausbremsen der Anwendung. Dies gilt natürlich sowohl für Linux als auch für Windows.

Die ersten Linux-Gehversuche Anfang 1996 bei DB0AAB-8 waren erfreulich. Den Anforderungen bezüglich Rechnerleistung wurde dort Rechnung getragen. Dort wurde seinerzeit ein 486DX2/66 mit 16 MB RAM und insgesamt 4,5 GB Plattenspeicher eingesetzt. Heutzutage ist gilt dies natürlich schon wieder als völlig veraltert. Es sollte sich hier auch zeigen, wie sich die Box im Zusammenhang mit einer sehr hohen Datenmenge verhält. Aus diesem Grund ist auch derzeit und bis auf weiteres jegliches Löschen von Nachrichten bei DB0AAB abgestellt. Falls die vorhandenen Platten voll sind, lässt sich die Kapazität problemlos auf höhere Kapazitäten erweitern.

Die Linux-Version der Mailbox hat vor dem ersten Einsatz im "rauhen" Betrieb bei DB0AAB auf der Fachhochschule München viele Versuche im stillen Kämmerlein über sich ergehen lassen. Trotzdem war es natürlich nicht möglich, alle Eventualitäten des praktischen Betriebs auszutesten, jedoch lief die Mailbox dort schon recht zügig auch unter höheren Belastungen sehr stabil. Inzwischen hat sich der Betrieb normalisiert und es gibt auch kein auffallendes Feedback von den Benutzern. Dies ist ein typisches Merkmal bei jeder technischen Entwicklung, dass Rückmeldungen stets nur im Fehlerfall kommen. Das ist zwar für den Entwickler nicht besonders erfreulich, aber im Sinne der Fortentwicklung durchaus wünschenswert und notwendig.

Generell gilt: Linux ist ideal über Funk zu bedienen und hat insgesamt Eigenschaften, die geradezu ideal für einen Einsatz der OpenBCM-Mailbox sind. Linux hat eigentlich nur einen einzigen Nachteil: Wer sich nicht mit UNIX auskennt, braucht ein relativ großes Grundwissen, um alle nötigen Kenntnisse zum Betrieb und zur Wartung eines Rechners zu haben.

3.3.1.2. Hardware requirements for Linux

Hier wird mindestens ein Rechner mit 80486-Prozessor bei 33 MHz, 8 MB RAM (besser 16 MB) und 1 GB Festplatte empfohlen. Mit schlechterer Hardwareausstattung ist die DOS-Version zu bevorzugen. Die Linux-Version sollte möglichst auf dem Mailbox-Rechner compiliert werden, um Schwierigkeiten mit inkompatiblen C-Bibliotheken zu vermeiden. Entsprechende Programme und Tools (C++-Compiler, Bibliotheken, etc.) sind bei jeder Linux-Distribution enthalten. Ggf. kann aber auch ein statisch gelinktes Binary verwendet werden.

3.3.2. Windows

3.3.2.1. Why Windows?

Für eine Anwendung wie eine PR-Mailbox ist Windows eigentlich kein allzu geeignetes Betriebssystem. Was die Vor- und Nachteile der einzelnen Systeme sind, sollte hier nicht diskutiert werden.

Tatsache ist, dass unter DOS nach wie vor nur die berühmten 640 kB Speicher vernünftig ansprechbar sind, und dass unter DOS kein besonders herausragendes Dateisystem zur Verfügung steht.

Windows NT/2000/XP hilft beiden Problemen ab. Es steht eine virtuelle Speicherverwaltung und das Dateisystem NTFS zur Verfügung.

Es ist nicht zu verschweigen, dass technisch die Lösung für Linux die weitaus bessere ist. Nur: Wer sich nicht mit UNIX auskennt, braucht ein relativ großes Grundwissen, um alle nötigen Kenntnisse zum Betrieb und zur Wartung eines Linux-Rechners zu haben. Windows ist hier ein wenig besser, vor allem ist das erforderliche Wissen etwas weiter gestreut. Dies ist die einzige Motivation, warum es die OpenBCM-Mailbox für Windows gibt. Es ist auch bei langem Nachdenken keine technische Begründung zu finden, warum Windows besser sein sollte als Linux, es ist eben nicht so. Da die Windows-Version der Mailbox nur wenig verbreitet ist, muss auch mit mehr bisher unentdeckten Fehlern als bei Linux gerechnet werden.

Die Windows-Version verhält sich sehr viel ähnlicher zur Linux- als zur DOS-Version. Das ist auch kein Wunder, da im 32 Bit-Mode vieles aus der DOS-Welt (z.B. Hardware- und BIOS-Zugriffe) nicht zur Verfügung steht, und auch z.B. die Speicherverwaltung eher vergleichbar mit Linux ist.

3.3.2.2. Which Windows: NT/2000/XP or Win95/98/ME?

Klare Antwort: Nur Windows NT/2000/XP! Die *bcm32.exe* läuft nur bedingt unter Windows 95/98/ME! Das liegt mehr oder weniger daran, dass man ohne Aufwand eine für beide Systeme kompatible EXE-Datei erzeugen kann, zum Betrieb mit einer Mailbox scheidet Windows 95/98/ME jedoch schon deshalb aus, weil es kein dem DOS überlegenes Dateisystem gibt. Mittlerweile ist Win95/98 und auch WinME als Betriebsystem langsam überholt und selbst Microsoft forciert die Nutzung eines stabileren Betriebsystems wie Windows XP. Unter Windows NT/2000/XP ist ausschließlich ein Betrieb mit NTFS-Dateisystem ratsam. Die Windows-Version der OpenBCM-Mailbox, BCM32, wurde ursprünglich unter Windows NT 4.0 mit Service Pack 5 entwickelt, mittlerweile wird unter Windows XP daran gebastelt.

3.3.2.3. Hardware requirements for Windows (NT/2000/XP)

Hier gelten dieselben Anforderungen an die Hardware wie unter Linux: mindestens ein schneller 486 oder ein PC der Pentium-Klasse sollte es sein. Zusätzlich sollte jedoch beachtet werden, dass Windows auf Grund der grafischen Oberfläche sehr viel verschwenderischer mit RAM umgeht als Linux. Unter WinNT sollten deshalb mindestens 32 MB, unter Win2000 oder WinXP mindestens 128 MB, besser 256 MB verwendet werden.

3.3.3. DOS

Zu DOS ist an dieser Stelle eigentlich nicht mehr viel zu erwähnen. Da das Betriebssystem DOS mittlerweile hoffnungslos veraltert ist, ist keinem Neueinsteiger zu empfehlen, diese Version zu installieren. Ggf. macht es bei Sinn, bestehende Systeme unter DOS weiterlaufen zu lassen, jedoch immer mit den Einschränkungen die dieses Betriebssystem mit sich bringt (640 kB Hauptspeicherbarriere, fehlerträchtiges FAT-Dateisystem, kein TCP/IP-Support...). Aber selbst hier macht es Sinn ein klein wenig Aufwand zu investieren, um die bestehende Mailbox auf Linux oder Windows umzurüsten (siehe hierzu auch Kapitel "22. Upgrade DOS version to Windows or Linux").
DOS – Ruhe in Frieden!

3.3.3.1 Hardware requirements for DOS

Um die geforderte Leistungsfähigkeit sicherzustellen, werden unter DOS folgende Eigenschaften vom verwendeten Rechner erwartet:

- Mindestens ein Rechner mit 80386-Prozessor
- Platte mit max. 20 ms Zugriffszeit und mindestens 500 kB/s Durchsatz
- Verwendung eines Platten-Cache Programms - deshalb sollte der Rechner mit mindestens 2 MB RAM bestückt sein (das entspricht dann 1,3 MB Cache), besser sind 4 MB; der Aufruf mit Standardeinstellung ist sinnvoll, also einfach in der AUTOEXEC.BAT "SMARTDRV 4000" eintragen (bei 4 MB RAM)
- Empfohlen wird MS-DOS 5.0 oder 6.2, mindestens jedoch DOS 3.3; abgeraten wird von DR DOS 5.0 oder MSDOS 6.0; auch der Einsatz von OpenDOS ist möglich (dieses ist für den privaten Einsatz kostenlos verfügbar). OpenDOS ist im Internet unter z.B. http://www.opendos.de/ zu finden.

3.4. Abstract

As an abstract of this chapter about the operation system and hardware comparations you can remember:
Linux is probably the best choice for the operation system. If necessary, you can also use the Windows version of OpenBCM. But today you should never starting using DOS as operation system for your mailbox.

The choice of correct hardware doesn't play a big role today. Effectively each PC bigger than 486 can be used running an OpenBCM mailbox.

4. Installation

4.1. Complete distribution of OpenBCM

If you are trying to install OpenBCM for the first time, it makes sense to download the complete distribution of OpenBCM. In this archive you will find all available files, also a lot of runutils etc. and a huge amount of example configuration files for OpenBCM.

When you are running OpenBCM using DOS, you need at least 2 files:

- *bcm.exe* (the main executable)
- *msg/messages.gb* (the english language file)

When you are running OpenBCM using Linux or Windows, you need at least only one file (the language file will be created by the executable if it does not exist):

- *bcm32.exe* (using Windows)
- *bcm* (using Linux)

A discription of all files of the mailbox you find in the end of this documentation. The most files are created by the mailbox process itself, if they are needed.

You can find a full release of OpenBCM (around 8,5 MB) in ham packet radio network in filesurf/FTP-Server at DB0FHN in Nuernberg/Germany, or in citizen packet radio band in filesurf at DBO274 in Oelde/Germany. And - for sure - there are a lot of other mailboxes in the packet radio network which are offering the full release of OpenBCM. So you can even search for mails with "fobcm" in the title of your home mailbox. For all people, who have internet access, you can also download the full release (and all other files) at http://dnx274.dyndns.org/baybox.

4.2. Where you will find updates

You can find each time the most newest update version, all available beta versions and also the complete documentation of OpenBCM...

-.in ham packet radio network in the filesurf/FTP-Server at mailbox DB0FHN-8 located in Nuernberg/Germany,
- in citizen band packet radio network in the filesurf at mailbox DBO274 located in Lippstadt/Germany,
- in the internet at http://dnx274.dyndns.org/baybox.

And - for sure - there are a lot of other mailboxes in the packet radio network which are offering the newest version of OpenBCM. In the ham packet radio network, normally the board BAYBOX@BAYCOM is used to exchange information, hints and the software itself. You will find for sure all release versions of OpenBCM in this board, maybe also beta versions.

4.3. Configuration of the harddrive

4.3.1. Using DOS

The mailbox will create for each mail a single file. When using DOS FAT16-filesystem you can only use max. 65535 cluster per partition. This result in a real storage amount of 8 kB per file when using a 540 MB harddrive (if you are using only one partition). Therefore it makes sense to create an own partion for usermail directory, bulletin directory and the system itself.

Since Windows 95B you can also use a 32bit-FAT filesystem. The advantage is, that only 4 kB harddrive space is used per file and the you can also create a partition bigger than 2 GB. But you have to take care of the licence restrictions of the software company and there may be problems with the DOS shell usage.

When using a 540 MB harddrive it's suggested to create following partitions:

Drive C: - first partition: 120 MB - System
Drive D: - secound partition: 300 MB - Bulletins
Drive E: - third partition: 120 MB - Usermails

When you are using a bigger harddrive you should calculate the space with an according factor.

4.3.2. Using Linux

You should minimum use the filesystem "ext2", maybe a raiser filesystem is also a good choice.

Normally ou can use all filesystems that are supported by Linux, but when you select the DOS filesystem "(V)FAT" you will loose a lot of performance speed.

The block size for the mail partition (*user* and *info* directory) should be 1024 Bytes.

4.3.3. Using Windows

Like DOS, you should avoid here also using the filesystem "(V)FAT". When using Windows NT/2000/XP it's recommended to use the filesystem "NTFS".

4.4. Installation using DOS

Die Mailbox kann so ziemlich unter jeder Konfiguration des Rechners laufen. Für einen sinnvollen Boxbetrieb sollten jedoch folgende Punkte erfüllt werden:

- Aktuelle DOS-Version (MSDOS 5.0 oder 6.2, IBMDOS 5.0 oder 6.3, Win95/98-DOS)
- Optimierte *config.sys*
- Optimierte *autoexec.bat*
- Sinnvolle Partitionierung der Platte

Zu einem ersten Start der Mailboxsoftware unter DOS sind mindestens zwei Dateien notwendig, und zwar *bcm.exe* (das DOS-Executable) und die Datei *msg/messages.gb* (englischsprachige Boxtexte).

Alle anderen zum Betrieb der Mailbox notwendigen Dateien und Verzeichnisse können von der Mailbox selbst erzeugt werden. Die Mailbox kann nun durch die Eingabe von "BCM <Enter>" gestartet werden. Damit sind erste Tests möglich.

An der Konsole kann jetzt ein Rufzeichen zum Login eingegeben werden, die Bedienung der Mailbox erfolgt anschließend wie über Funk gewohnt.

Durch Drücken der Tastenkombination ALT-X wird die Mailbox beendet.

Um die Mailbox wirklich "on air" betreiben zu können, müssen diverse Dateien angepasst werden (siehe dazu auch Kapitel "5. Configuration for the first start" und folgende).

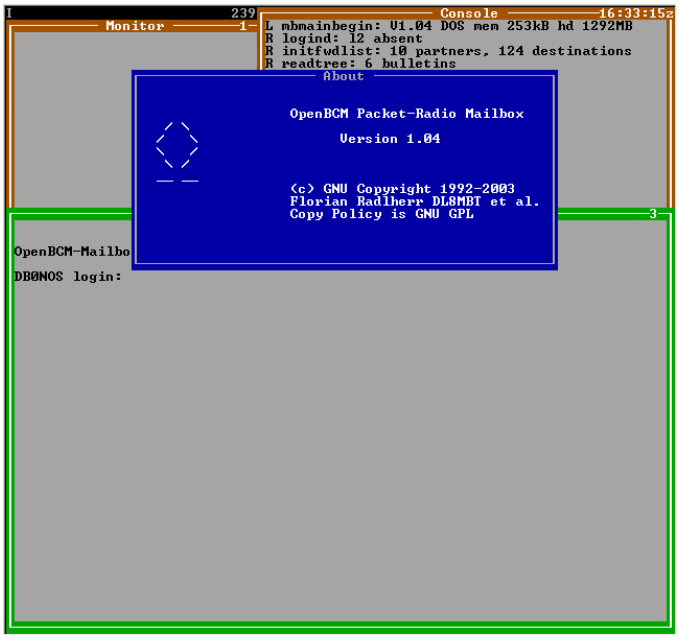


Fig. right: Screenshot of the good old DOS interface

4.4.1. Optimised *config.sys*

In der Datei *config.sys* sollte ein Wert FILES=50 eingestellt werden. Außerdem sollte DOS auf maximalen freien konventionellen Speicher konfiguriert sein (mittels DOS=HIGH, EMM386, etc.). Nicht wirklich für den Mailboxbetrieb notwendige Gerätetreiber sollten nicht geladen werden, notwendige mit "DEVICEHIGH=" in den hohen Speicherbereich. Sinnvoll ist der Betrieb auf einer Monochrom-Grafikkarte ("Hercules") oder auf einer CGA-Karte, möglichst nicht EGA oder VGA, da man hierdurch mehr Arbeitsspeicher gewinnt. Sofern es das BIOS zulässt, ist auch ein Betrieb ohne Grafikkarte möglich.

Beispiel aus einer *config.sys* für Monochrom-Karten:

```
DEVICE=C:\DOS\EMM386.EXE NOEMS I=A000-AFFF I=B800-EFFF
```

Beispiel aus einer *config.sys* für EGA/VGA-Karten:

```
DEVICE=c:\DOS\EMM386.EXE NOEMS I=C800-EFFF
```

Werden SCSI-, RLL- oder ESDI-Plattencontroller verwendet, so ist die Option "X=C800-CFFF" und ein Aussparen dieses Bereichs in den "I="-Anweisungen erforderlich, da diese Contoller in diesem Bereich ein BIOS haben.

Beispiel einer vollständigen *config.sys*:

```
LASTDRIVE=Z
DEVICE=C:\DOS\HIMEM.SYS
DOS=HIGH,UMB
BUFFERS=50,3
FILES=60
DEVICEHIGH=C:\DOS\EMM386.EXE NOEMS I=A000-AFFF I=B800-EFFF
REM For monochrom video only I=A000-AFFF
SHELL=C:\DOS\COMMAND.COM /P
```

4.4.2. Optimised *autoexec.bat*

Wenn die Mailbox fix auf einem Rechner installiert wird, sollte sie in der Datei *autoexec.bat* in einer Endlosschleife aufgerufen werden. Als Tastaturtreiber sollte nur der Original-DOS-Treiber *keyb.com* verwendet werden. Andere Tastaturtreiber führen oft zu sehr merkwürdigen Effekten, auch wenn sie Arbeitsspeicher einsparen.

Beispiel einer *autoexec.bat*:

```
@ECHO OFF
SET TEMP=C:\TEMP                *** Verzeichnis muss existieren!
SET COPYCMD=/Y                  *** für MSDOS 6.x: Keine Rückfragen beim copy
SET FLEXNET=C:\FLEXNET          *** Pfad für PC/FlexNet einstellen
PATH C:\DOS;C:\BCM;C:\BCM\RUN   *** Im Suchpfad sollten sich nur
                                    wirklich notwendige Pfade befinden
PROMPT $P$G                     *** Übliches Befehlsprompt
LOADHIGH SHARE.EXE              *** Regelt Zugriffsrechte auf der Platte
LOADHIGH SMARTDRV 4000          *** Plattencache in den hohen Speicherbereich
LOADHIGH KEYB GR                *** Deutscher Tastaturtreiber
:LOOP
CHOICE /T:j,5 Knoten starten?   *** MSDOS 6.x: Abfrage ob Knoten starten
IF ERRORLEVEL=2 GOTO ENDE
ECHO Knoten wird gestartet...
CALL SFLEX.BAT                  *** Vor der Mailbox die FlexNet-Treiber starten
ECHO Knoten aktiv
SCANDISK /ALL /AUTOFIX /NOSAVE /NOSUMMARY /MONO C:
                                *** Platte ausmisten (MSDOS 6.x),
CD \BCM
ECHO Box wird gestartet...
BCM                             *** Startet die Mailbox
CALL STOPFLEX                   *** Beendet FlexNet
GOTO LOOP                       *** Wenn BCM beendet, dann wieder zurück
:ENDE
```

Anmerkung: Beim Aufruf von *smartdrv* sollte in der Kommandozeile sämtlicher vorhandener XMS-Speicher angegeben werden. Die OpenBCM-Mailbox nutzt XMS-Speicher nur als Plattenpuffer, eine anderweitige Nutzung ist nicht möglich. Auch dies bringt allerdings einen sehr großen Geschwindigkeitsvorteil.

Es sollte darauf geachtet werden, dass nach dem Start der OpenBCM-Mailbox noch mindestens 100 kB, besser aber mehr als 250 kB konventioneller Speicher frei sind. Dies ist normalerweise zu erreichen.

4.5. Installation using Linux

4.5.1. System requirements for Linux

When using Linux you need a Linux kernel 2.0.0 or higher. You should select a stable kernel, it is not recommended to use a hacker kernel. Good experiences are made with kernel 2.2.x- and 2.4.x. ELF support and the required libraries are needed.

4.5.2. Main installation using Linux

- Normally when using Linux, you will have first to compile your mailbox by yourself. This procedure is described in detail in chapter "27.2. Using Linux".
- For the installation of the mailbox program you will need "root" rights. Therefore you should operate really carefully because you can damage a lot in your system if you do not know what you are doing there. Any to liability is for sure assumed. The following procedure is only a suggestion. When you have your own experiances you may install in a similar way.
- First you should create a system user and a system group, both called "bcm". Principially you don't have to care about UID and GID. When using a network system they should be used unified, but that's not a requirement. It's suggested to use UID 666 and GID 66. If your system is using UID or GID already, it's no problem to choose another ID. The creation of an user "bcm" will help in administration of the file access rights. You can also use the program *bct* as login shell for that user. This will result in an automatic start of *bct* if you will log into your system as user "bcm".
- For the main mailbox directory it's suggested that you use */bcm*. It's also possible to use another directory of your flavour as well, but then you should use an enviromental variable $BCMHOME for that directory. If you use */bcm*, you don't have to care about the enviromental variable. For the start of your mailbox it's recommended to use the start script *startbcm*. *startbcm* is a shell script, that will define the correct enviromental variable $BCMHOME, goes into that directory and will start *bcm*.
- If your mailbox is running unmanned, you should create an entry in file */etc/inittab*. This entry should look like:
  ```
  bc:23:respawn:/bcm/startbcm
  ```
  Through this entry your mailbox will be started automatically after next system reboot and will also be started again if the software has crashed or the sysop has killed the linux process or terminated the mailbox through SHUTDOWN command (e.g. for updating to a new version). After executing the script *startbcm* the system is waiting some time before a new start is tried not to create to much error messages in respawn process if a real failure is occuring (e.g. *bcm* has been deleted). The

runlevel (the numbers after bc:) must apply to your system installation. There are different versions using different linux operating systems, nevertheless the state "Multiuser with network" must be reached.

- While all files are installed (or are copied from existing DOS/Windows version), they should belong to user "bcm" and group "bcm". Just to be sure you should execute in directory *bcm* and maybe also in directories *user* and *info*:
  ```
  chown -R bcm:bcm *
  chmod -R u+rw *
  ```
- While the mailbox is normally not running as root process you need a root login for the sysop when he wants to administrate the system using the mailbox SHELL command. For this, you can use the program *pw*. This program needs to have root access, means UID root. Therefore you should execute for this program:
  ```
  chown root.root pw
  chmod 6777 pw
  ```
  After executing this, the program *pw* is able to give a sysop root rights. Alternatively you can run a second installation of OpenBCM, complete with root rights, to use it for the system administration. When doing this, you have to configure in the second OpenBCM other ports for the TCPIP access in file *init.bcm* and also another port or device in file *init.l2* to avoid conflicts.

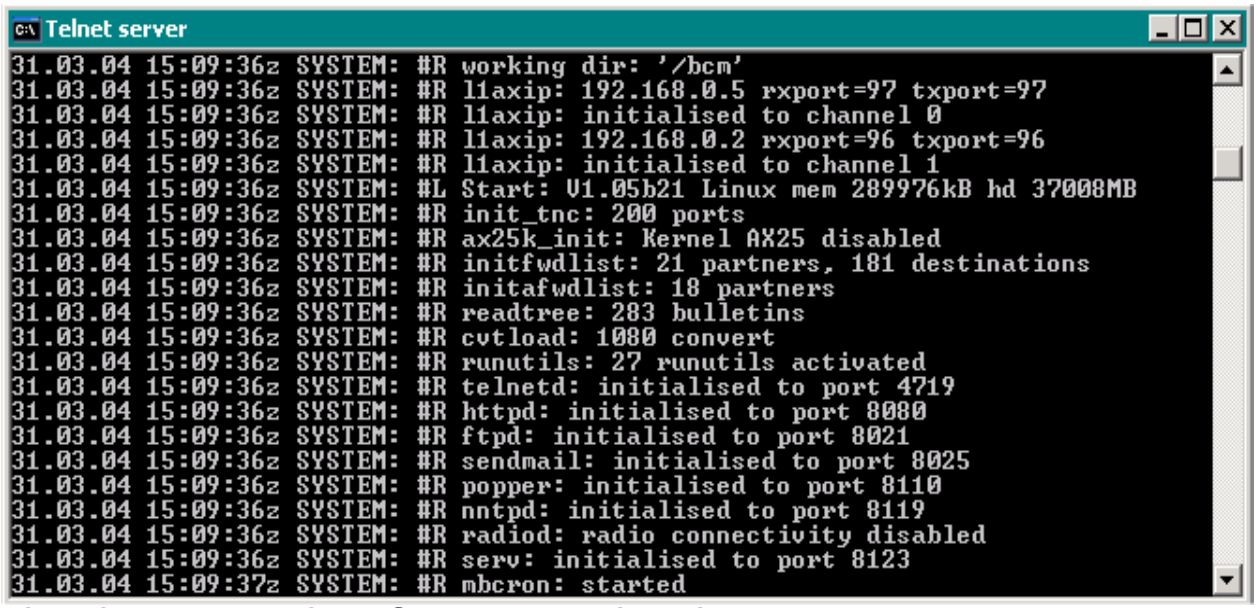


Fig. above: Screenshot of OpenBCM running Linux

To use the mailbox really "on air", you have to configure some files (see chapter "5. Configuration for the first start" and following).

4.5.3. Login using TCP/IP running with Linux

OpenBCM is waiting as default for a telnet connection on port 4719. This port number 4719 is free to change, but it is recommended to use the default value of 4719 as unification, if there are no problems with other running applications. You can change the port number with parameter "TELNET_PORT <port>" in file *init.bcm*.

If you setup a value of 0, the telnet port is disabled.

You can also use the simple terminal program *bct* for a login. But this is only working if you define a service named "bcm" in file */etc/services*:

```
bcm 4719/tcp # OpenBCM-Mailbox
```

You can also add a similar line for UDP access in this file, but it is not really needed.

4.5.4. Login using Linux

While Windows version of OpenBCM has no real user interface itself, you can only handle the mailbox through an external program.

You can use the simple terminal program *bct* to login into your mailbox. If this program is running you may are not allowed to change the screen resolution. You can terminate this program with a logout using "q" or while pressing CTRL and C together. To start the *bct* program you have to type in:

bct [<hostname>]

If your OpenBCM and the bct program is running on the same machine you don't need to add the hostname as an option. When bct and mailbox are running on different machines you have to add the hostname of the PC where the mailbox is running. After starting the bct program a login prompt is shown.

If the hostname of that PC where the terminal connect to the mailbox will come from is added in file *rhosts.bcm*, only the callsign (no password) is asked when logging in and the logged in user has also automatic sysop state in the mailbox. Have also a look at chapter "5.7. Remotehost (rhosts.bcm)" for more information about this.

To login into the mailbox you can also use telnet. Please keep in mind that a telnet program is sending each character that you type in, immediately. So you may get "special effects" when using the backspace or delete key while sending commands to the mailbox. The command to make a telnet connect looks like this:

telnet <hostname> bcm

or, if you have not defined an entry "bcm" in */etc/services*:

telnet <hostname> 4719

4.6. Installation using Windows

4.6.1. System requirements for Windows

OpenBCM is only compatible with Windows NT, 2000 or XP, not Windows 95/98/ME or even Win3.1x! Additionally your Windows version must be installed including the TCPIP protocol for windows networks. Without this protocol OpenBCM for Windows will not work.

4.6.2. Main installation using Windows

For a first run of OpenBCM you need at least the file *bcm.exe* (the main executable). You should copy this file to an empty folder, preferable *c:\bcm*.

When you execute this file the first time a lot of subdirectories and configuration files are created, which are used in the further application run. With this, first steps are possible.

You can now login via telnet or via a packet terminal program using Flex32/XNET.

If you press the keys ALT and X in the mailbox window together, the mailbox will shutdown.

To use the mailbox really "on air", you have to configure some files (see chapter "5. Configuration for the first start" and following).

In order that the mailbox is automatically restarting after a crash (normally this will never happen), you can use a small software tool "OpenBCM Watchdog". It is recommended to install this tool, if you use the mailbox at an exposed place for an uncontrolled run.

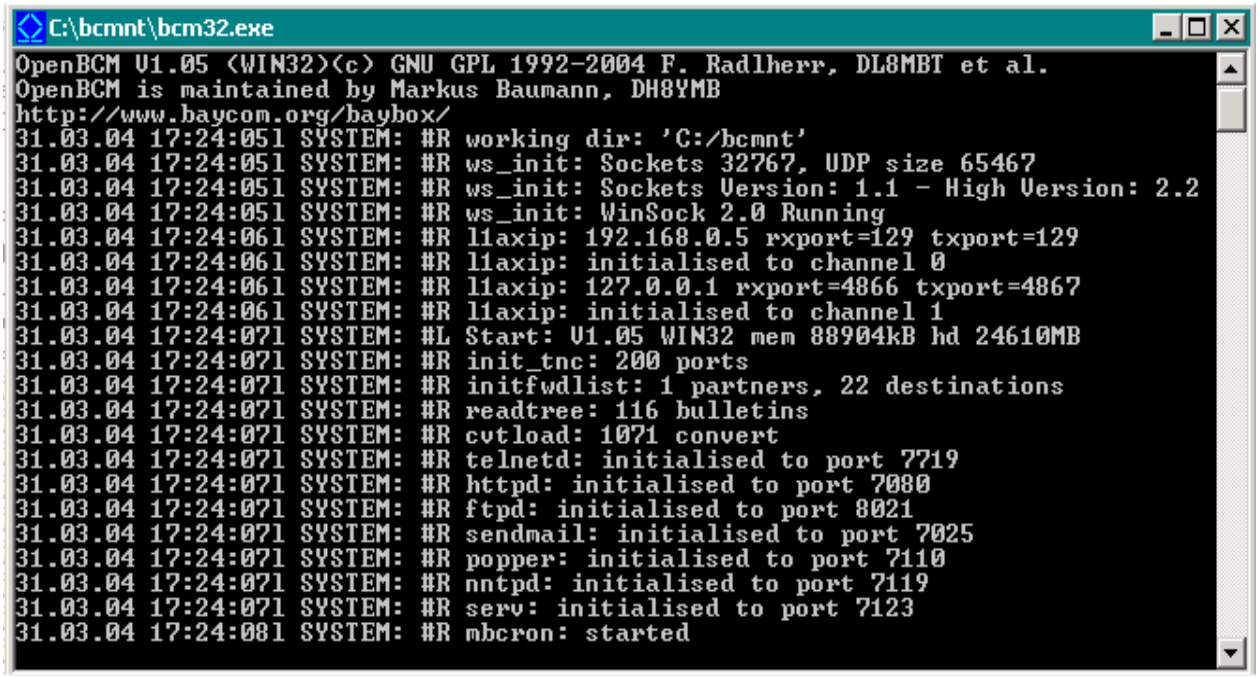


Fig. above: Screenshot of OpenBCM running Windows NT/2000/XP

4.6.3. Login using TCP/IP running with Windows

OpenBCM is waiting as default for a telnet connection on port 4719. This port number 4719 is free to change, but it is recommended to use the default value of 4719 as unification, if there are no problems with other running applications. You can change the port number with parameter "TELNET_PORT <port>" in file *init.bcm*.

If you setup a value of 0, the telnet port is disabled.

4.6.4. Login using Windows

While Windows version of OpenBCM has no real user interface itself, you can only handle the mailbox through an external program.

For the first login to your mailbox you should use the windows telnet program.

To start a telnet connect you can use following DOS shell command (e.g. click START/EXECUTE and type in):

```
telnet localhost 4719
```

Please keep in mind that a telnet program is sending each character that you type in, immediately. So you may get "special effects" when using the backspace or delete key while sending commands to the mailbox.

Instead of using telnet, you can also use an external packet radio program like Paxon, Winstop or WPP ... . All these programs can use Flex32 and/or XNET with an internal AXUDP link which is connected to the OpenBCM mailbox.

4.7. Update to a newer software version

When using DOS you should update your OpenBCM version best executing a batch file. It makes sense to create a file *newbcm.bat* (EXTRACT feature in *config.h* must be defined while compiling the DOS version, or you need the sysop tool *extract.exe* by DF3VI) with following contents:

```
extract baybox %1.p0
7plus %1.p01
copy bcm.exe bcm.alt
if exist %1.exe copy %1.exe bcm.exe
del %1.*
```

If 7plus.exe reports an error, you make have to request a **.cor* file.

The update process is now really simplified:

```
    oshell newbcm bcm121
    shutdown
```

When using Windows or Linux, the update process is much easier. Here you can start a shell with OSHELL command and type in all commands you want to replace the existing bcm executable with a new one. Running Linux you can even start a fresh compilation of the current source files (simply use "make install"). When you have only remote access to your mailbox PC you should consider to start a second process in the background before manipulation your OpenBCM installation. If the update process will be not successful, you may have a second chance to get the mailbox back to life.

5. Configuration for the first start

You learned in the last chapters about the requirements of operation systems and hardware. So, now it is time to start the configuration of your OpenBCM mailbox. For the first steps, it is ok, that you may have no radio connection, but later on, you should think about that.

What do you have to configure?

- main config (*init.bcm* and maybe *init.l2*)
- bulletin boards list and descriptions (*bulletin.bcm* and *boardinf.bcm*)
- sysop password (*passwd.bcm*)
- timing file (*crontab.bcm*)
- forward file (*fwd.bcm*)
- definition of autosysop (*asysop.bcm*)
- lanuguage definition file (*speech.bcm*)
- beacon configuration (*beacon.bcm* and *beachead.bcm*)
- reject and hold definitions (*reject.bcm*)
- automatic board convertion (*convert.bcm*)
- automatic distributor convertion (*convat.bcm*)
- runutil definitions (*runutil.bcm*)
- remote host configuration (*rhosts.bcm*)

5.1. Main configuration (*init.bcm* and *init.l2*)

The main configuration is saved in file *init.bcm*. This file is initially created after the first start of your OpenBCM. All parameters of this file can be configured by command at the mailbox prompt when the mailbox is running and will be saved automatically. Only if your mailbox is not running you should/can change parameters by an external text editor. Following procedure is recommended for the first start of OpenBCM:

- when using Linux or Windows it is recommended to edit file *rhosts.bcm* before the first start
- start your OpenBCM mailbox
- log into your mailbox: when using Linux/Windows use a telnet connect ("telnet localhost 4719"), whne using DOS you can easily login in the windows by input of your callsign and <ENTER>
- type in command like "BOXADDRESS db0aab.#bay.deu.eu" (surely you should better use your own box address!)
- select the login/forward callsign of your mailbox with command "MYCALL db0aab-8 db0aab-7"
- type in command "MKBOARD / TMP" to create the file *bulletin.bcm* with a default setting (only needed if *bulletin.bcm* does not exist already)
- enable your mailbox with command ENABLE
- now end your mailbox process with command SHUTDOWN

At this moment the file *init.bcm* is created, while the own hierarchical address and box callsign is already configured. These are the most important settings. Now you should check following parameters in file *init.bcm*:

boxheader
The "advertising banner" – maybe locator, sysop callsign, town of BBS, e.g.: "boxheader JN78ov Zwettl OP:OE3DJB"

sysopcall
sysop callsign (needed for login at the panel)

infopath
directory, where all bulletin mails are saved, maybe you can select here an own data partition of your harddrive

userpath
directory, where all usermails are saved, maybe you can select here an own data partition of your harddrive

userlife
default lifetime of usermails, maybe you should increase this value

dosinput
only needed using DOS; defines if your mailbox is started with or without shroom (select with shroom: 0, without shroom: 1)

userpw
allow users to create their own password (userpw 1) or disallow this (userpw 0)

When you are using Windows or Linux you should also edit the file *init.l2*, which configures the layer2 connection to the outside world.

The files *init.bcm* and *init.l2* are described in detail in chapters "24.4. Initialization file *init.bcm*" and "24.5. Initialization file *init.l2*".

5.2. Bulletin boards (*bulletin.bcm* and *boardinf.bcm*)

In contrast to FBB or MSYS mailbox system OpenBCM knows bulletin boards. Mails which do not fit into the defined board structur will be put into the (temporary) board TMP. New boards are only created automatically, if parameter CREATEBOARD is set to 1.

Following is a short example for a file *bulletin.bcm* used in ham radio:

```
; OpenBCM-Mailbox Bulletin Listfile
; Syntax: Board Max.Lifetime [Min.Lifetime]
all       120  10
 alle      30   5
 todos     30   5
 tous      30   5
modes     360  90
 amtor    360 360
 aplink   360   0
 atv      360   0
 clover   360   0
 cw       180   0
 fax      360   0
 gtor     360   0
 pactor   360   0
 rtty     360   0
 sattv     90  90
 sstv     360 360
[...]
weather    10   5
 meteo     10   5
 wx-info  365 365
z           5   5
```

In *bulletin.bcm* you are not only defining the board structure, but also the default, maximum and minimum lifetime of your boardmails.

Normally you will have a huge amount of board names. You can create main boards and subboards, that are according to main boards. In the above example you will find a main board "modes" and a lot of subboards that correspond to this main board. The structur of main and subboards was only introduced for the user to get a clear arrangement of boards. Often users like to use boards which are similar the same (e.g.: 6m, six, 50mc, 50mhz) – you can sum up those boards to one board easily by using *convert.bcm* (have a look at chapter "5.11. Automatic board conversion (convert.bcm)" for this). Have always a look at your TMP board to see if it is necessary to create a new board.

In file *boardinf.bcm* you can optinally define description of your boards. These descriptions will be shown when listing a board with command DIR or LIST. The syntax is following:

```
BOARDNAME Description
```

If no description is defined for a board in *boardinf.bcm*, it is clear, that nothing can be shown to the user. Hint: use capital boardnames for your main boards and lower case boardnames for your subboards.

A short example of a *boardinf.bcm* file:

```
ALL       Bulletins addressed to all users (english language)
alle      Bulletins addressed to all users (german language)
cept      Info about european administration CEPT
[...]
WEATHER   Topic Weather and weather reports in general
wefax     Weather fax pictures
```

5.3. Sysop password (*passwd.bcm*)

A password for the sysop can be entered in the file *passwd.bcm*. Without this password, no remote sysop activity is possible. In the first line of this file you can define the password for the BayCom procedure, the second line defines the password for MD2, and in the third line you can define the MD5 password.

5.4. Timing (*crontab.bcm*)

The mailbox will execute different proceses at fixed times. The point of starting time of these processes can be specified in the file *crontab.bcm*. The timing is closely relative to *crontab* known by UNIX operating systems.

A simple file *crontab.bcm* is created automatically after the first start of the mailbox. You should additionally add some routines for data backup. You can find a description in detail of this file in chapter "15. Timed process".

5.5. Store & Forward (*fwd.bcm*)

IMPORTANT NOTE: Please read this chapter carefully! The correct configuration of your forward is important to the automatic spreading of mails and is the most important job of a sysop! Have also a look at chapter "9. Complete forward configuration" for more information about configuration of *fwd.bcm*.

In file *fwd.bcm* you can define in which manner bulletin and user mails are forwarded to other mailbox systems. Unfortunately this is often misconfigured. The result are often stucked mails or unnecessary ping-pong forward of mails.

Later in this description you can find in detail how the forward configuration file is working. At this place you learn with the help of some examples how to configure a good working forward definition.

Let us start with a simple example: DB0BCM-8 is you own new mailbox and you would like to create a simple bulletin forward. The sysop of station DB0AAB-8 is asked and prepared to do forward with your mailbox. Only bulletins with director @DL should be exchanged between the two systems, e.g. MEINUNG@DL, IBM@DL, SATTV@DL etc.

The forward file *fwd.bcm* of DB0BCM-8 will look like:

```
DB0AAB - DB0AAB-8
 DL
```

At station DB0AAB-8 (which is also running OpenBCM) the following entry can be found in file *fwd.bcm*:

```
[...]
DB0BCM - DB0BCM-8 DB0AAB
 DL
[...]
```

In the first line of a forward block you define the callsign of your forward partner, the timetable when forward should run (normally you define "-" or 24 times an "A" here) and information for the connect path to the forward partner.

In the example for DB0BCM-8, the target callsign is "DB0AAB" (there is no need for a SSID). The mailbox DB0BCM can directly connected the user port of the digipeater DB0AAB, therefore the connect path for DB0AAB-8 is simply DB0AAB-8.

For the station DB0AAB-8 the first line of the forward block is a little bit more complex: DB0AAB-8 can connect DB0BCM-8 only "via" DB0AAB, because the mailbox DB0AAB-8 has no direct access to the userport of DB0AAB. Therefore the connect path is here "DB0BCM-8 DB0AAB".

In the second line of a forward block, you can define the directors which should be used for that forward partner. Those lines must start with a space as first character. In both examples above, you see the director DL, this means all mails with "...@DL" will be forwarded.

Confused at the moment? You can connect to other OpenBCM mailboxes and have a look at there forward configuration with command "DIR PATH". Most times this help a lot. But you can also discuss a correct connect path for your needs with other sysops.

Now, let's go on:
In the example above only mails with the director "@DL" are exchanged. Now also the directors "@BAYCOM" (which is used for OpenBCM software distribution) and "@EU" should be used.

This will lead at DB0BCM-8 to this forward block:

```
DB0AAB - DB0AAB-8
 BAYCOM DL EU
```

...and at DB0AAB-8:

```
[...]
DB0BCM - DB0BCM-8 DB0AAB
 BAYCOM DL EU
[...]
```

The changes are small, only the directors are added.

Up to now it doesn't care if the mailbox DB0BCM-8 is not working – only bulletin mails are saved and all these mails can also be read in DB0AAB-8, because the mails have been copied and not deleted in the source mailbox. Now we want to add the handling of usermails. The sysop has to assume the responsibility: If his mailbox receives an usermail, to forward partner will delete the successful forwarded usermail. If the own mailbox is wrong configured, this usermail will stuck and is maybe lost for the packet mailbox network. Therefore only receive usermail when you believe your can handle your forward configuration well!

Because you are unsure at the moment, you have decided that you would only receive usermails for users of your own mailbox DB0BCM-8, but you would not receive usermails for other mailboxes than DB0BCM-8, and you would send all other usermails from your mailbox to DB0AAB-8. DB0BCM is located like DB0AAB in Bavaria (Bayern). The compete hierarchical addresses are "DB0AAB.#BAY.DEU.EU" respectively "DB0BCM.#BAY.DEU.EU" and so very similar. In *fwd.bcm* of DB0AAB-8 only the entry DB0BCM must be added to the forward block:

```
DB0BCM - DB0BCM-8 DB0AAB
 DB0BCM BAYCOM DL EU
```

At DB0BCM-8 all useful addressed usermails must be send to DB0AAB-8. Therefore all continents (except Europe), all european countries (except Germany), all german regions (except Bavaria) and all bavarian mailboxes (except of your own, DB0BCM) must be added to the forward block of DB0AAB-8 in *fwd.bcm*.

You find this is inconvenient? Only at first view. If you got later a second forward partner, you have simply to move useful entries from the first to the new forward block. It does not make sense to send usermail twice or more to different forward partners.

The file *fwd.bcm* at DB0BCM-8 looks now as following:

```
DB0AAB - DB0AAB-8
; Bulletin directors
 BAYCOM DL EU
; Usermails: continents (except Europe)
 .AF .AFRC .AS .AU .AUST .CEAM .CARB .MDLE .NA
 .NOAM .OC .OCEA .SA .SOAM
; Usermails: countries in Europe (except Germany)
 .AUT .BEL .BGR .BIH .CHE .CZE .DEU
 .DNK .ESP .EST .FIN .FRA .GBR .GIB
 .GRC .HRV .HUN .IRL .ITA .LTU .LUX
 .LVA .MKD .MLT .NLD .NOR .POL .PRT
 .ROM .RUS .SVK .SVN .SWE .TUR .UKR
 .YUG
; Usermails: German regions (except Bavaria)
```

```
 .#NRW .#BLN   .#RPL  .#BRB   .#SAA
 .#BW  .#SAR   .#HB   .#SAX   .#HES   .#SLH
 .#HH  .#THR   .#MVP  .#NDS
; Usermails: mailboxes in own region (except own mailbox)
 DB0AAB DB0ABH DB0AHO DB0AST DB0BOX DB0FFB DB0FP
 DB0FSG DB0GAP DB0IGL DB0IRS DB0ISW DB0KCP DB0KFB
 DB0KP  DB0LAN DB0MAK DB0MAS DB0MFG DB0MWS DB0PV
 DB0RGB DB0SL  DB0ULM DB0WGS DB0ZB  DB0ZKA DF0ANN
 DK0MUC DK0MUN
```

In chapter "9. Complete forward configuration" you can find a more detail description of hierarchival address, shortcuts of continents, countries and regions. The above example complies the experience of the author.

DB0BCM-8 has been now successful tested since some weeks and this mailbox should be now connected via an own link to digipeater DB0AAB. Thus, DB0BCM-8 is connected to an own RMNC digipeater. Additionally WPROT-Info should be exchanged between DB0AAB-8 and DB0BCM-8. This is configured with entry "$WP". The new forward files look now as following:

At DB0AAB-8:

```
[...]
DB0BCM - DB0BCM-8 DB0AAB
 DB0BCM BAYCOM DL EU $WP
[...]
```

At DB0BCM-8:

```
DB0AAB - DB0AAB-8 DB0BCM
; Bulletin directors
 BAYCOM DL EU
; WPROT info
 $WP
; Usermails: continents (except Europe)
 .AF .AFRC .AS .AU .AUST .CEAM .CARB .MDLE .NA
 .NOAM .OC .OCEA .SA .SOAM
; Usermails: countries in Europe (except Germany)
 .AUT .BEL .BGR .BIH .CHE .CZE .DEU
 .DNK .ESP .EST .FIN .FRA .GBR .GIB
 .GRC .HRV .HUN .IRL .ITA .LTU .LUX
 .LVA .MKD .MLT .NLD .NOR .POL .PRT
 .ROM .RUS .SVK .SVN .SWE .TUR .UKR
 .YUG
; Usermails: German regions (except Bavaria)
 .#NRW .#BLN   .#RPL  .#BRB   .#SAA
 .#BW  .#SAR   .#HB   .#SAX   .#HES   .#SLH
 .#HH  .#THR   .#MVP  .#NDS
; Usermails: mailboxes in own region (except own mailbox)
 DB0AAB DB0ABH DB0AHO DB0AST DB0BOX DB0FFB DB0FP
 DB0FSG DB0GAP DB0IGL DB0IRS DB0ISW DB0KCP DB0KFB
 DB0KP  DB0LAN DB0MAK DB0MAS DB0MFG DB0MWS DB0PV
 DB0RGB DB0SL  DB0ULM DB0WGS DB0ZB  DB0ZKA DF0ANN
 DK0MUC DK0MUN
```

The entry "$WP" and for the connect path a "via DB0BCM" has been added, because the mailbox DB0BCM-8 has to connect now the target via the RMNC-Digi.

Now an additional forward partner OE5BCM-8 should be added to DB0BCM-8. All received bulletins and all usermails for Austria (except region Tirol) should be send to this mailbox. Between DB0AAB-8 and DB0BCM-8 the directors "@WW" and "@EU" should be additionally exchanged. OE5BCM-8 will send only usermail for DB0BCM-8. Additionally the director THEBOX should be added. With this director normally only mails in board F are exchanged. This board is – as all one-letter-boards – a sysop only board. Board F is used for exchange sysop information, years ago the director THEBOX was also used for white-page-info exchange with obsolet DieBox systems.

fwd.bcm at DB0AAB-8:

```
[...]
DB0BCM - DB0BCM-8 DB0AAB
 DB0BCM BAYCOM THEBOX DL EU WW $WP
[...]
```

fwd.bcm at OE5BCM-8:

```
[...]
DB0BCM - DB0BCM-8 OE5BCM
 DB0BCM BAYCOM THEBOX DL EU WW $WP
[...]
```

fwd.bcm at DB0BCM-8:

```
DB0AAB - DB0AAB-8 DB0BCM
; Bulletin directors
 BAYCOM THEBOX WW EU DL
; WPROT info
 $WP
; Usermails: continents (except Europe)
 .AF .AFRC .AS .AU .AUST .CEAM .CARB .MDLE .NA
 .NOAM .OC .OCEA .SA .SOAM
; Usermails: countries in Europe (except Germany)
 .BEL .BGR .BIH .CHE .CZE .DEU
 .DNK .ESP .EST .FIN .FRA .GBR .GIB
 .GRC .HRV .HUN .IRL .ITA .LTU .LUX
 .LVA .MKD .MLT .NLD .NOR .POL .PRT
 .ROM .RUS .SVK .SVN .SWE .TUR .UKR
 .YUG
; Usermails: mailboxes in austrian region Tirol
 .#OE7.AUT
; Usermails: German regions (except Bavaria)
 .#NRW .#BLN   .#RPL  .#BRB   .#SAA
 .#BW  .#SAR   .#HB   .#SAX   .#HES   .#SLH
 .#HH  .#THR   .#MVP  .#NDS
; Usermails: mailboxes in own region (except own mailbox)
 DB0AAB DB0ABH DB0AHO DB0AST DB0BOX DB0FFB DB0FP
 DB0FSG DB0GAP DB0IGL DB0IRS DB0ISW DB0KCP DB0KFB
 DB0KP  DB0LAN DB0MAK DB0MAS DB0MFG DB0MWS DB0PV
 DB0RGB DB0SL  DB0ULM DB0WGS DB0ZB  DB0ZKA DF0ANN
 DK0MUC DK0MUN
OE5BCM - OE5BCM-8 DB0BCM
; Bulletin directors
 BAYCOM THEBOX WW EU DL
; Usermails
 .AUT
```

Note: In the DB0AAB forward block the entry ".AUT" has been replaced with".#OE7.AUT".

How can you verify your configuration? Run your mailbox and have always a look at the command UNKNOWN. You will get the content of the file *trace/unknown.bcm* with this command, where all mails are listed which got stuck in your mailbox. Here you can find all useless and wrong addressed mails, but also all mails which got stuck because of a wrong forward definition.

In chapter "9. Complete forward configuration" you can find a more detail description of forward configuration.

5.6. Autosysop (*asysop.bcm*)

The file *asysop.bcm* enables automatic sysop status on certain ports just after login. The entry of a password is not necessary.

Note: a wrong configuration of this file can be a security hole, so be sure what you are doing here! If the file *asysop.bcm* does not exist this function is deactivated. Simply delete this file if you are unsure!

Syntax of *asysop.bcm*:

```
callsign uplink downlink
```

Example:

```
DJJ812 none none                (directly connectable)
DJJ812 DBO812-2 DBO812-2        (if the SSID of entry port is -2)
```

If the up- or downlink should contain nothing, "none" must be declared. Downlink is the call of node or digipeater, that is closest to the BBS. Uplink is the call of node, that is the farthest from the BBS, as well that node over that the users gets into the packet network. Each line must end with a carriage return.

Notice:
The uplink can be easily falsified, downlink theoretically never, since it is usually the own one. However you should take care that your downlink can newer changed from outside. When connecting from your own node your uplink is identical with your downlink, so take care of your SSID (when using Flexdigi 3.3e it can be changed from outside, Flexdigi 3.3g is more save, because a new password procedure is implemented here).

- When using TCPIP the uplink keywords are HTTP, NNTP etc. – these keywords can also be used.
- If uplink and downlink are declared both "none", it is a direct connect without node or digipeater between it.
- If a digi owns the SSID 0 or no SSID, the call is valid without SSID.

5.7. Remotehost (*rhosts.bcm*)

The file *rhosts.bcm* was shortly introduced in chapter "4. Installation". This file defines which IP numbers/hostnames are secure. When login with an IP number/hostname defined here, only the callsign is asked and the user has automatically sysop state.

If the uplink computer can not be found in *rhosts.bcm*, a TTY password (TTYPW) must be entered for login and the user will not get sysop state automatically. This means, only user with a valid TTY password can login into the BBS.

Note: Also the own IP address and the hostname "localhost" must be included in *rhosts.bcm*, if a login without password is desired for the local PC. After a fresh installation and when doing a first login, it is important to add the own IP address/hostname in this file to get access to the BBS.

Be aware: The own computer should be only added to file *rhosts.bcm*, if it is sure that only sysops have access to this computer. If you are using for example WAMPES or AX.25 utils with AXSPAWN command on the same computer, everybody could login as sysop with an unsecure *rhosts.bcm* file – so in this case you should not add your own computer to file *rhosts.bcm*.

5.8. Multilanguage configuration (*speech.bcm*)

In file *speech.bcm* the assignment between the language files (*msg/messages.**) and callsign prefix patterns, who should use a special language file as default, are defined.

You can chose freely how much callsign patterns are assigned to a language file, but the patterns must be devided through a space and the line does not exceed more than 80 characters.

The syntax of a line is like following:

```
MSG-pattern callsign-pattern
```

Example:

```
DL DG DH DL DO OE
```

In above example the language "DL" will be default used by callsign prefix patterns "DG", "DH", "DL", "DO" and "OE".

You can check the all current assignments of the BBS with command "a s" – they are listed in the brackets.

A special feature is a callsign prefix pattern (e.g. "DL") which is added in brackets. Then HELP-, CTEXT-, QTEXT-, etc. –files of this language are used in the BBS, but for all online messages the selected language is used.

If a callsign prefix pattern is not declared, the language "GB" will be used as default.

The first line of files *msg/messages.** are containing a description with version info of the language file, which is also shown when using the command "a s".

If the file *speech.bcm* is missing while starting OpenBCM software, a default file will be created – dependent on parameter CALLFORMAT in *init.bcm* either for ham radio or citizen band usage.

5.9. Beacon configuration (*beacon.bcm* and *beachead.bcm*)

The file *beacon.bcm* defines the target callsign for the beacon transmission.

Syntax for DOS:

```
<mycall> <tocall> [ <viadigi> ... ]
<mycall> <tocall> [ <viadigi> ... ]
```

Syntax for Linux/Windows:

```
<tocall> [ <viadigi> ... ]
<tocall> [ <viadigi> ... ]
```

The parameters have following meaning:

```
<mycall>    own callsign that should be used for the beacon
<tocall>    target callsign (UI address) of the beacon
<viadigi>   digis that should be used for the transmission
```

Example for DOS (when using Linux or Windows you have simply to cut off the mycall "DB0AAB-8"):

```
DB0AAB-8 MAILS DB0AAB              beacon will be transmitted via DB0AAB
DB0AAB-8 MAILS DB0AAB OE7XAR       beacon will be transmitted via OE7XAR
```

If you would like to send a fixed text in each beacon, you have to save this text in a file *beachead.bcm*. You can of course use macros in this text file.

In chapter "17. Mail beacon" you will find more info regarding the beacon.

5.10. Reject and Hold (*reject.bcm*)

In file *reject.bcm* you can define rules how to proceed with received mails. The mails can be rejected or they can be set on hold (means no further forward to other forward partner) for some time. Comments can be used everywhere, but must start with a leading ";".

The exact syntax of a reject line in *reject.bcm* is:

```
<action> [<sender] [>dest] [$bid] [@at] [ .B | .P ] ;comment
```

Meaning of:

- <action>:
  R = Mail is rejected
  G = Mail is rejected, if entered locally without AX25-PW
  E = Mail is rejected, if entered locally without AX25/TTYPW
  F = Mail is set to hold, if forward is not using a password

H = Mail is set to hold
L = Mail is set to hold, if entered locally
P = Mail is set to hold, if entered locally without AX25-PW
O = Mail is set to hold, if entered locally without AX25/TTYPW

- [<sender] callsign of sender
- [>dest] receiver (board or callsign)
- [$bid] a BID/part of a BID
- [@at] Address (Director or mailbox callsign)
- .B bulletin mail
- .P usermail

The actions E and O can be only used with Linux/Windows version!

You can use "?" (any character) and "*" (more than one character or no character) as wildcards. You can use "*" even at the beginning or inside a keyword. You can use regular expression for the keyword, see also chapter "34.5. Regular expressions" for this.

You can negotiate a condition with a leading "!", for example a line

```
R @DB0IRS >!DH3MB
```

would reject all mails for DB0IRS which are not directed to DH3MB (not useful).

You can use also an OR-condition, for example a line

```
G .B <!(DH8YMB,DL9CU
```

would reject all bulletin mail which are entered locally without a password verified login, except of mail from DH8YMB and DL9CU. You can also use an OR-condition without a negotiation, but this normally makes no sense.

Handle the *reject.bcm* feature very carefully, for example a line

```
R <*CB*
```

would reject more mails as expected first and would lead to a lot of angry people!

Some more examples:

```
; reject citizen band callsigns
  R <CB?CB
; reject FLOHMARKT mails
  R >FLOHM*
; reject worldwide MEINUNG mails (they should use board DEBATE for this!)
  R >MEINU* @WW
; df0ar can only send mails to OE3DZW
  R <^DF0AR$ >!^OE3DZW$ ...
; reject some citizen band mailbox BIDs
  R $*DBO123*
  R $*NL3DGH*
; hold all bulletins without AX25 password (check parameter HOLDTIME for this!)
  P .B
; reject all bulletins without AX25 password
  G .B
; reject all bulletins without AX25 password, except those from stations DB0* or DH8YMB
  G .B <!(^DB0,^DH8YMB
```

One line can contain more than one condition. Only if all conditions are met, the action is executed.

If the option "REJECTEDIT" is compiled in your executable mailbox file, you can use the internal REJECTEDITOR for editing the file *reject.bcm* easily.

With parameter HOLDTIME the sysop can configure the time how long mails are set to hold status if they met a hold criterium. An useful value for HOLDTIME is "48" (means 2 days). You can check with command DIR HOLD all mails that are on hold state. If a sysop has checked a held mail, he can use the command "hold -u <board> <nr>" or "f -h <board> <nr> @ <new director>" to put the mail back into forward.

In chapter "12. PW & Hold/Reject functions" you can find more info about file *reject.bcm*.

5.11. Automatic board conversion (*convert.bcm*)

The file *convert.bcm* can be used for an automatic board conversion dependency of the rules in this file. Also the maximum lifetime can be changed. In former BCM mailbox systems the files *convname.bcm* and *convlife.bcm* are used for this. These two files are not used any longer, so you can delete them without a problem.

Comments can be used everywhere, but must start with a leading ";".
A single rule line without a comment must not exceed more than 54 characters, with comment and including CR/LR it must not exceed more than 80 characters.

Characters with special meaning: < @ ~ $ % &

<callsign	Callsign of the sender
@address	mailbox or director
~Word	searches complete words in the title
text	searches for occurance of "text" in the title

The characters % $ & can only be used without an argument.

$	received by S&F
%	user input
&	user input without password

Important: only use spaces and no TABs as separator. You can use more than one rule in a line, which is interpreted as an "and" condition. To use them as "or" conditions you can simply use a second rule line. All conditions must be met to execute the rule. Small or capital letters are irrelevant.

A word must be surrounded by a space, this means ~NO = " NO ", or if this word exist in the title beginning: "NO ", or if this word exist at the end of the title " NO".

LT 000 means, that the maximum lifetime of *bulletin.bcm* is used. The maximum lifetime of *bulletin.bcm* can be changed with *convert.bcm* higher or lower. A lifetime of 0 can not be achieved with *convert.bcm*, a lifetime of 1 should be enough, but only if the minimum lifetime of *bulletin.bcm* is not violated. The order of the entries is very important, because only the first match is executed.

When using <, @, and text, you can use regular expressions (see also chapter "34.5. Regular expressions"). This is not possible when using ~.

Example of a file *convert.bcm*:

```
;Example convert.bcm
;from    to        LT mode/scan title   ; comment also possible here
;                                                 <max. line length
BAYBOX   BAYBOX   000 $                ; do not convert if received via S&F
F        F        000 @THEBOX          ; do not convert if address is THEBOX
*        Z        002 <DL3XYZ          ; each bulletin from DL3XYZ to "Z"
DUTCH    RTTY     014 <PA3ESF          ; board DUTCH from PA3ESF to RTTY
*        Z        002 @CBCALL          ; invalid h-address to Z with LT 2
*        AMSAT    180 @AMSAT           ; all @AMSAT to AMSAT
;*       AMSAT    180 @ AMSAT           ; wrong, no blank after @, <, or ~
6M*      SIX      090 $                ; $ = if received from a box
50*      SIX      090 $                ; $ = if received from a box
TV*      SATTV    180 %                ; % = if received from a user
*        H        002 &                ; & = if from a user without PW
*        H        002 & @WW            ; &    "     and @WW
*        H        002 & @EU            ; &    "     and @EU
; title scan examples:
*        PROBLEM  014 ?                ; all questions to PROBLEM with LT 14
*        CONTEST  014 ~CONTEST         ; ~ means word match
*        Z        002 ~NO TEXT         ; if word NO and "TEXT" matches
```

```
*        SUCHE    000 WANT              ; if "WANT" anywhere in title
*        BILDER   000 JPG               ; if JPG in title
DX*      QSLMGR   000 QSL
; regular expression examples (see also Help REGular_expressions):
*        Z        002 ^R:99             ; title starts with "R:99"
*        SUCHE    000 ^S:               ; title starts with "S:"
; 'and'-combination of blank separatet items:
*        SUCHE    000 ? QSL             ; if "?" and "QSL" in title
*        DIPLOME  360 $ @AMSAT ~AWARD DX ; all items must match
; only LT conversion:
KEPLER   *        014 $                 ; if received via S&F
;KEPLER   *        014                  ; wrong, that does not work
KEPLER   *        014 MIR DAY           ; title depending
; only to-field conversion:
CBMAIL   Z
ASCII    BILDER
JPG      BILDER
; wildcard for multi matching:
IMAGE*   BILDER
DX*      DXNEWS                        ; min. 2 chars before *
; end of example file
```

Everybody can have a look of the complete currently used file *convert.bcm* by typing the command "convert -a". If a keyword is added, only the lines which apply to this keyword are shown. The keyword can be used as regular expression.

If the option "DF3VI_CONV_EDIT" is compiled in your executable mailbox file, you can use the internal CONVEDITor for editing the file *convert.bcm* easily.

5.12. Automatic director conversion (*convat.bcm*)

The file *convat.bcm* can be used for conversion of invalid directors. The conversion is used for local mails. If a mail, which applies the conversion rules, is received by forward the mail is not changed but the new director is handled internally!

Example:

```
; Syntax: <old director> <new director>
OEDL DL
DLOE DL
OEBA DL
BAOE DL
ALLE DL
ALL  WW
EURO EU
EUR  EU
AUT  OE
DEU  DL
```

If a user sends locally a mail with "s meinung@oedl", this mail is saved and forwarded like it is addressed "MEINUNG @ DL". If in the forward procedure a mail with "INFO @ EUR" is received, this mail will be forwarded to other partners like it is addressed @EU. So, you do not need entries like EUR or OEDL in file *fwd.bcm.* Note: The director of a mail, which is received by forward, is not changed, the new director is only internally handled!

When you type in the command CONVAT you can have a look at the current file *convat.bcm*.

5.13. Runutils (*runutil.bcm*)

To make an external program available for an user, the name of the program must be added to file *runutil.bcm*. The execution of the runutil is the same like the sysop execution with command OSHELL.

The file *runutil.bcm* has following syntax (Linux example):

```
;command  program             options  comment
BIN*READ  binread                      compressed reading of mails
MEM*INFO  "cat /proc/meminfo" -qc      show memory usage of PC
PSL*INUX  "ps aux"                     show all linux processes
FILENAME  filename            -qs      search filename of a mail    (SYSOP)
FM*AIL    fmail               -s       Generate a sysop F mail      (SYSOP)
USERSTAT  runutils/userstat   -s       Users statistic              (SYSOP)
READSTAT  runutils/readstat            Board statistic
FL*OG     runutils/flog                S&F Statistic digis (flog <yymmdd>)
UL*OG     runutils/ulog                User statistic (ulog <yymmdd>)
QTH       qth                 -I       calculate a locator
NEWS      news                         News and dates
SYSINFO   runutils/sysinfo             Systeminfo
7*MAIL    7mail               -faD     send file a 7plus mail     (FILESURF)
7G*ET     7get                -faD     readout with 7plus         (FILESURF)
7C*OR     7cor                -faD     generate 7plus COR file    (FILESURF)
BS*GET    bsget               -faD     readout autobin splitted   (FILESURF)
V*IEW     fileview            -faD     liste a file archiv        (FILESURF)
I*NFO     fileinfo            -faD     read file description      (FILESURF)
F*IND     filefind            -faD     search file in description (FILESURF)
N*EW      filenew             -faD     list new files by date     (FILESURF)
XD*IR     filexdir            -faD     extended "DIR"             (FILESURF)
FSE*DIT   fsedit              -fasD    add file description (SYSOP,FILESURF)
FSD*EL    fsdel               -fasD    del file description (SYSOP,FILESURF)
SYS7M*AIL sys7mail            -fasD    7Mail no limits      (SYSOP,FILESURF)
HILFE     fshelp              -faD     additional Filesurf help   (FILESURF)
```

The fields have following meaning:

- command
  This is the command in the mailbox which must be typed in when the runutils is executed. The letters before "*" must be entered to execute the command, the letters after "*" are optional for the command execution.
- program
  This is the external Linux/DOS/Windows program, that should be executed. If the shell command contain a space, the program name must include " characters.
- options
  This field is optional. The field contains of single letters with a leading "-". The letters have following meaning:
  - -S: can only executed by sysops.
  - -F: can only executed in filesurf mode.
  - -Q: file *rundat.bcm* is not generated when executing.
  - -C: user parameters are not used when executing.
  - -I: while the runutil is executed, further input of the user is not send to the runutil, but to the mailbox (can only be used in Linux or Windows version)
  - -P: can only executed, if the user is logged in with password
  - -D: enviroment variables (like using DPBox) are set for the runutil (can only be used in Linux or Windows version)
  - -T: characters ">", "<" and "|" can be used and are send to the runutil (can only be used in Linux version)
- Comment
  A short description of the runutil (< 80 characters).

Additionally following rules must be followed:

- A comment line start with a leading ";".
- Fields are separated with one or more blanks.

6. Radio connection

Auf einem Rechner kann je nach Betriebssystem neben der Mailbox auch noch andere Packet Radio-Software laufen. Außerdem können natürlich mehrere Rechnersysteme mit verschiedener Software laufen, die dann miteinander verbunden werden. Beispielhaft seien hier einige Kombinationen genannt:

- DB0AAB: An der FH-München laufen Digi und Mailbox jeweils auf einem eigenen Rechner. Die beiden Systeme sind via Ethernet miteinander verbunden. Der Knoten läuft unter DOS mit PC/Flexnet, der Mailboxrechner läuft unter Linux und dient gleichzeitig noch als TCP/IP-Server. Zusätzlich läuft noch eine Sprachmailbox, die ebenfalls am Ethernet-Bus hängt.
- OE1XLR/OE1XAB: Hier dient ein Rechner sowohl als Knoten als auch als Digi. Der Funkbetrieb erfolgt mit PC/Flexnet und USCC-Karten.
- OM0NVA: Bei diesem System ist die Mailbox über die serielle Schnittstelle des Rechners mit einem RMNC/Flexnet-Knoten verbunden.

Es sind natürlich noch viele weitere Konfigurationen denkbar. Im Folgenden soll aber hauptsächlich beschrieben werden, wie man die OpenBCM-Mailbox mit dem Digipeater verbindet.

Der Anschluss an die Außenwelt ist unter DOS mit PC/Flexnet möglich. Durch die Verwendung von Flennet gibt es nur einen "breiten" Datenkanal auf dem die Kommunikation zwischen Box und Flennet erfolgt. Damit gibt es nicht die vom Hostmode her bekannten Beschränkungen auf irgendwelche statischen TNC-Kanäle. Auf welchem Funkport die Ausgabe erfolgt, hat für die Mailbox keine Bedeutung.

Unter Linux erfolgt der Funkanschluss mit einem integrierten Layer2 oder mit dem Linux-Kernel-AX.25.

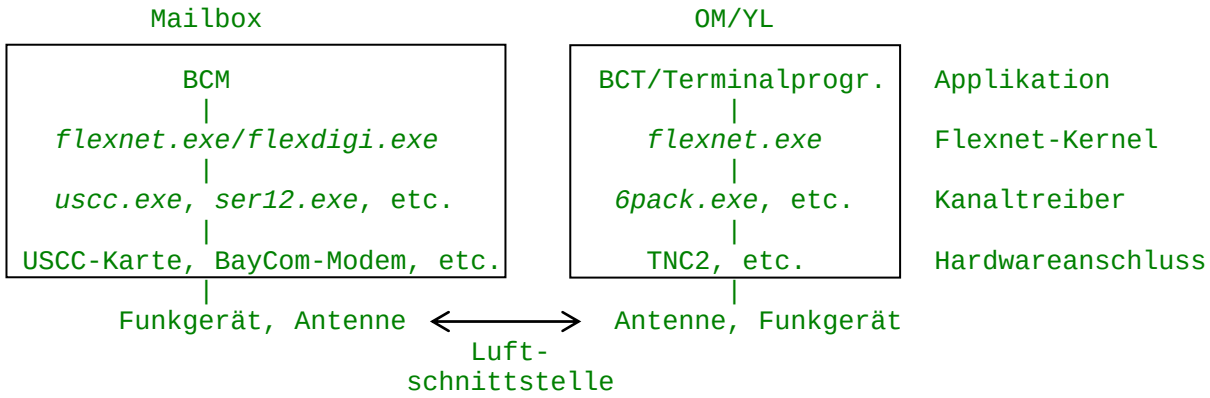
Unter Windows ist ebenfalls der Betrieb mit dem integrierten Layer2 möglich.

Außer den genannten, gibt es noch eine Reihe weiterer Konfigurationsmöglichkeiten. Wer glaubt eine interessante Lösung geschaffen zu haben, kann dies in einer Mail an BAYBOX @ BAYCOM schreiben.

6.1. Radio connection using DOS with PC/Flexnet

Unter DOS haben alle Konfigurationen eines gemeinsam: Die Mailbox wird auf PC/Flexnet "aufgesetzt". Je nachdem, ob PC/Flexnet gleichzeitig als Digi dienen soll oder nicht, muss *flexdigi.exe* gestartet werden. Die Kommunikation nach außen wird immer von PC/Flexnet gesteuert, jede unter PC/Flexnet einsetzbare Funk-Hardware kann also auch bei der DOS-BayCom-Mailbox verwendet werden.

Grafisch lässt sich diese Kommunikation z.B. folgendermaßen darstellen:



Die Daten werden von der OpenBCM-Mailbox an den geladenen Flexnet-Kernel übergeben. Dieser gibt die Daten an die Kanaltreiber weiter.

Im Kanaltreiber werden diese Daten je nach Hardware aufbereitet und an der entsprechenden Schnittstelle ausgegeben (z.B. serielle Schnittstelle). Im Modem werden diese Daten in über das Funkgerät übertragbare Signale umgewandelt, die

Antenne strahlt die Packet-Radio Daten ab. Am anderen Ende der Verbindung befindet sich das Funkgerät des Users, die Daten gehen den umgekehrten Weg. Das BayCom-Terminal (oder irgendein anderes Terminal) stellt diese Daten am Bildschirm dar, der OM oder die YL freut sich über die neuen Nachrichten in der Mailbox.

Im Rechner muss neben dem Betriebssystem (DOS) folgendes laufen:

- Der oder die Kanaltreiber
- Der Flexnet-Kernel (evtl. mit Digi-Erweiterung)
- Die OpenBCM-Mailbox

Die Hardware bestimmt, welche Kanaltreiber zu laden sind. Es wird folgende Hardware unterstützt:

- BayCom-1k2-Modem (*ser12.exe*)
- BayCom-Parallel-9k6-Modem (*par96.exe*)
- BayCom-(U)SCC-Karte (*uscc.exe*)
- VANESSA-Karte (*vanessa.exe*)
- RMNC-CRC-KISS-Rechnerkopplung (*kiss.exe*)
- TNC2 (*6pack.exe*)
- IPX (*ipxpd.exe* bzw. *ipxn.exe*)
- AXIP (*ippd.exe*)
- Ethernet-Multicast nach BPQ (*ether.exe*)
- div. Soundkarten (*psadrvcr.exe*, *dsk.exe*, *dsk50.exe*, *dg1scr.exe*, *wss.ewe*, *wss_9k6.exe*, *sblast.exe*)

PC/Flexnet ist ein unter vielen Entwicklern verteiltes Projekt, in den Mailboxen sind in der Rubrik FLEXNET (manchmal auch RMNC) immer wieder neue Treiber zu finden.

Nachdem der Rechner gebootet wurde, ist zuerst der Flexnet-Kernel *flexnet.exe* zu laden. Anschließend wird, je nach dem, ob der Rechner gleichzeitig auch als Netzknoten dienen soll, auch *flexdigi.exe* geladen. Jetzt folgen die Kanaltreiber. Die Reihenfolge bestimmt die Port-Nummer (0..14), ist jedoch ansonsten unwesentlich. Sind alle Treiber geladen, wird Flexnet mit *flex.exe* aktiviert. Ab diesem Zeitpunkt ist Flexnet aktiv. Evtl. können oder müssen mit *fset.exe* verschiedene Parameter gesetzt werden. Zum Abschluss wird die OpenBCM-Mailbox *bcm.exe* gestartet. Mit dem Befehl VERSION kann jetzt überprüft werden, ob Flexnet erkannt wurde. Wird das Programm *flexdigi.exe* geladen, so wird im Versionsbefehl "FlexDigi" angezeigt, ansonsten "FlexNet" für Flexnet ohne Node.

Zu beachten ist, dass *flexdigi.exe* nicht frei verteilt wird, aber unter Angabe des Netzknoten-Rufzeichens bei Gunter, DK7WJ@DB0ZDF.#RPL.DEU.EU angefordert werden kann.

Womit Leute immer wieder Probleme haben: Mit *flexdigi.exe* wird die interne Struktur komplett umgebaut. Normalerweise hängen Applikationen wie *bcm* via Flexnet-Kernel direkt an den L1-Kanälen. Mit *flexdigi.exe* hängen sie an einem (virtuellen) Kanal 15 und kommunizieren nur über diesen mit der Außenwelt.

Dadurch sieht von der Parametrierung her gesehen FLEXDIGI mit BCM genauso aus wie z.B. RMNC/Flexnet (oder PC/Flexnet) mit BCM auf getrennten Rechnern. Man braucht im FLEXDIGI also auch einen Link-Eintrag zur Box auf Kanal 15. QSOs laufen immer über den Digi, was den Durchsatz etwas bremst und mehr Speicherplatz benötigt.

Daher die Empfehlung, FLEXDIGI wegzulassen, wenn keine Netzknotenfunktion gebraucht wird. Auch HOP2HOP-digipeating wird seit Version 3.3e vom Flexnet-Kernel unterstützt (siehe *flexnet.doc* und "FSET /?").

Für Sysops, die FLEXDIGI einsetzen, ist die gedruckte Sysop-Dokumentation für RMNC/Flexnet und PC/Flexnet erhältlich bei:

```
FlexNet-Gruppe Darmstadt
Gunter Jost, DK7WJ
Lichtenbergstr. 77
D-64289 Darmstadt
Deutschland
```

Achtung: Flexnet ist im CB-Funk nicht erlaubt! Daraus folgt, dass die DOS-Version der OpenBCM im CB-Funk nicht eingesetzt werden kann!

6.1.1. Configuration

Flexnet zeichnet sich durch nur wenige notwendige Einstellungen aus. Für jeden Kanal sind jeweils nur der MODE und das TXDELAY einzustellen, bei manchen Kanaltreibern ist auch dies nicht notwendig.

6.1.1.1. Configuration with node

Wird *flexdigi.exe* geladen, so werden die notwendigen Einstellungen der Datei *flexnet.fpr* entnommen, sie brauchen nicht bei jedem Start neu eingestellt zu werden. In dieser Datei werden neben dem TXDELAY und MODE für jeden Kanal auch das "ssid" sowie die eingetragenen Links gespeichert. Beim ersten Start von *flexdigi.exe* wird eine Defaultkonfigurationsdatei *flexnet.fpr* erzeugt. Diese kann entweder mit *fset.exe* oder mit dem Miniterminal *tnc.exe* den Erfordernissen angepasst werden.

Der erste Start von Flexnet könnte etwa so aussehen:

```
Eingabe                          Bemerkung
-------------------------------------------------------------------------
LOADHIGH FLEXNET.EXE 150         Flexnet-Kernel, 150 kB Puffer
LOADHIGH FLEXDIGI.EXE            Flexnet-Digi
LOADHIGH USCC.EXE /P=0x300 /I=5  Kanaltreiber für USCC-Karte
LOADHIGH SER12.EXE 2             BayCom-Modem an der COM2
FLEX.EXE                         Aktivierung von FlexNet
TNC.EXE                          Starten des Mini-Terminals
ESC C                            Connect zum Digi "FLXNET-0"
MYCALL DB0AAB 0 15               Einstellen des Digi-Calls, SSIDs 0-15
MODE 0 9600trzums                Einstellen der Kanal-Modes für den
MODE 1 1200cums                  Ports 1-4 der USCC-Karte (entspricht
MODE 2 19200dtrz                  den Flexnet-Kanälen 0-3)
MODE 3 19200trz
MODE 4 1200c                     ...und für das BayCom-Modem
PAR SSID 0 0                     Einstellen der Kanäle mit SSID
PAR SSID 1 1                     (= keine Exklusivlinks)
PAR TXD 10 0                     Einstellen der TX-Delays der
PAR TXD 25 1                     verschiedenen Kanäle
PAR TXD  5 2
PAR TXD  5 3
PAR TXD 25 4
MAIL DB0AAB-8                    Einstellen der lokalen Mailbox
Q                                Verbindung zum Knoten beenden
ESC Q                            Beenden des Terminals
Y                                Ende bestätigen
BCM.EXE                          OpenBCM-Mailbox starten
(...)                            OpenBCM-Mailbox bedienen
FLEX.EXE /U                      Flexnet aus dem Speicher entfernen
```

Vor- und Nachteile dieser Lösung sind:

- (+) Wenig Stromverbrauch, billig (Leistung pro Rechner etwa 50 W, das sind 438 kWh (etwa 60 Euro) pro Jahr.
- (-) Es steht weniger RAM für die Box zur Verfügung.
- (-) Stürzt der Digi ab, so stürzt auch die Box mit ab und umgekehrt.
- (-) Insgesamt schlechtere Stabilität sowohl von Box als auch von Digi. Die Zielsetzung beider Systeme ist unterschiedlich (Digi: Schnelle

Interruptreaktionszeit, Box: Viele Dateizugriffe), so dass hier mit Konflikten zu rechnen ist.

Beispiel einer *sflex.bat* (allgemein):

```
LOADHIGH FLEXNET.EXE 150
LOADHIGH FLEXDIGI.EXE
LOADHIGH XXX.EXE (entsprechender L1-Treiber, z. B. SER12.EXE oder USCC.EXE)
FLEX.EXE
```

6.1.1.1.1. Mailbox and digi using one PC with USCC cards

- (+) Günstige Lösung, mittlere Übertragungsraten möglich, max. 38,4 kBit/s je Kanal.

Beispiel *sflex.bat* (mit USCC-Karten):

```
LOADHIGH FLEXNET.EXE 150
LOADHIGH FLEXDIGI.EXE
LOADHIGH USCC.EXE /P=0x300 /I=7
FLEX.EXE
```

6.1.1.1.2. Mailbox and digi using one PC with Baycom modem

- (+) Geringster Hardware-Aufwand, gute Lösung für erste Tests.
- (-) Nur 1200 Bit/s Übertragungsgeschwindigkeit, nur 1 Kanal.

Beispiel *sflex.bat* (mit BayCom-Modem):

```
LOADHIGH FLEXNET.EXE 150
LOADHIGH FLEXDIGI.EXE
LOADHIGH SER12.EXE 2
FLEX.EXE
```

6.1.1.2. Configuration without a node

Da es in diesem Fall keine Parameter-Datei gibt, müssen die Parameter MODE und TXDELAY nach jeder Aktivierung mit dem Programm *fset.exe* eingestellt werden.

Der erste Start sieht ähnlich aus:

```
Eingabe                          Bemerkung
-------------------------------------------------------------------
LOADHIGH FLEXNET.EXE 80          Flexnet-Kernel, 80 kB Puffer
LOADHIGH KISS.EXE 2              Kanaltreiber für KISS auf COM2
                                 (z.B. Kopplung mit RMNC-Knoten)
FLEX.EXE                         Aktivierung von Flexnet
FSET.EXE MODE 0 19200c           19k2 mit RMNC-CRC
BCM.EXE                          BayCom-Mailbox starten
(...)                            BayCom-Mailbox bedienen
FLEX.EXE /U                      Flexnet aus dem Speicher entfernen
```

Sowohl das Laden, als auch das Parametrieren von Flexnet geschieht in der Startdatei *sflex.bat*. Diese Datei wird aus der *autoexec.bat* mit "CALL SFLEX.BAT" aufgerufen.

Befindet sich die Flexnet-Software im Verzeichnis *c:\flex*, so kann die Startdatei *sflex.bat* etwa folgendermaßen aussehen:

```
C:
CD \FLEX
LOADHIGH FLEXDIGI.EXE 150
LOADHIGH FLEXNET.EXE
LOADHIGH <Kanaltreiber>.exe <Parameter für Kanaltreiber>
FLEX.EXE
FSET.EXE TXD 0 10
FSET.EXE MODE 0 1200c
```

Eventuell können neben der Applikation *bcm.exe* auch andere Applikationen geladen werden. Unter DOS müssen diese Applikationen TSR's (Hintergrund-Programme) sein, unter einer anderen Oberfläche (Windows, OS/2, etc.) ggf. auch Vordergrundprogramme.

6.1.1.2.1. Connection to node system via ethernet

- (+) Sehr schnelle und fehlersichere Übertragung
- (+) Es können auch mehr als 2 Rechner miteinander verbunden werden (z.B. auch eine Sprachmailbox und/oder ein TCP/IP-System)
- (-) Kosten von 2 Ethernetkarten (ca. 20 € pro Karte + Abschlusswiderstände)

Beispiel *sflex.bat* (mit Novell-Netz):

```
CALL NOVELL.BAT (Lädt die Novell-Treiber)
LOADHIGH FLEXNET.EXE 50
LOADHIGH IPXN.EXE
FLEX.EXE
```

...oder eine Lösung mit Public-Domain-Treibern, hier eine Beispiel *sflex.bat* (mit PD-Packet-Treiber):

```
CALL ETHERNET.BAT (Lädt den Ethernet-Packet-Treiber)
LOADHIGH FLEXNET.EXE 50
LOADHIGH IPPD.EXE -I:0x60
FLEX.EXE
```

6.1.1.2.2. Connection to node system via serial device

- (+) Billig, da Schnittstelle bei Mainboards oft vorhanden
- (+) Kopplung auch mit anderen Systemen (z.B. RMNC) möglich
- (+) Einfache Konfiguration
- (-) Gefahr von Übertragungsfehlern wenn kein CRC eingeschaltet ist (deshalb CRC-Kiss an beiden Enden der Verbindung und möglichst 16550 UARTs verwenden)
- (-) Es können jeweils nur zwei Rechner miteinander verbunden werden, ein Bus-System wie bei Ethernet gibt es nicht

Beispiel *sflex.bat* (KISS):

```
LOADHIGH FLEXNET.EXE 50
LOADHIGH KISS.EXE /P=0x3F8 /I=4
FLEX.EXE
FSET.EXE MODE 0 19200c
```

6.1.1.2.3. Connection to node system with USCC cards

Hier erfolgt die Verbindung über zwei USCC-Karten mit TTL-Pegeln. In früheren Konfigurationen war diese Lösung empfehlenswert. Da inzwischen aber alle gängigen Systeme mit CRC-gesichertem KISS-Mode ausgestattet sind, ist eine Drahtkopplung über USCC eher nicht empfehlenswert, denn:

- Der RMNC ist im KISS-Mode schneller als mit HDLC (bis 115 Bit/s bei 12 MHz).
- Serielle Schnittstellen sind sehr billig und mit FIFO (16550A) erhältlich.
- RS232 hat vernünftige Schnittstellenpegel für Leitungen bis ca. 20 m, TTL ist nicht vernünftig zu übertragen.
- USCC-Karten sollten also anstatt für Drahtlinks ausschließlich für Funklinks eingesetzt werden, dafür sind sie konzipiert.

6.2. Radio connection using Linux

Wie bei PC/Flexnet gibt es auch unter Linux kaum Funk-Hardware, die nicht unterstützt wird. Die meisten für DOS genannten Hardware-Kombinationen sind deshalb auch unter Linux denkbar.

Unter Linux existiert außerdem eine Menge an Packet Radio-Software, dazu gehören Knotensystem, Mailboxen, DX-Cluster, TCP/IP-Server und Convers-Systeme.

Eine Verbindung dieser Systeme untereinander ist über Pipe-Devices mit KISS oder über Loopback-AXIP realisierbar. Kernel AX.25 wird unterstützt. Es muss aber gesagt werden, dass auch unter Linux die Ressourcen nicht unerschöpflich sind. Laufen bei einem PR-Knoten viele Dienste parallel, sollten diese ggf. auf mehrere Rechner verteilt werden.

Unter Linux existiert außerdem die ausgereifte Knoten-Software XNET, außerhalb des Linux-PC kann aber auch PC/Flexnet oder RMNC/Flexnet verwendet werden.

6.2.1. Radio connection with integrated Layer2

Für Packetbetrieb ist in der OpenBCM-Mailbox für Linux die Funktion des Layer2 enthalten. Dieser wird über die Datei *init.l2* konfiguriert. Nur wenige Parameter müssen eingestellt werden. Eine Beispieldatei befindet sich im Kapitel "24.5. Initialization *file init.l2*".

Der AX.25-Teil der Box kann zwei verschiedene Interfaces ansteuern:

- Zum einen funktioniert KISS-Mode an einer seriellen Schnittstelle oder an einem Pseudo-TTY zur internen Verbindung mit einer anderen Software. Wird eine serielle Schnittstelle angesteuert, so kann über den MODE- Befehl die Baudrate eingestellt werden. Diese ist jedoch nur zur Startzeit einstellbar (also vorher *init.l2* editieren) und im laufenden Betrieb nicht mehr änderbar.
  WICHTIG: Der KISS-Port ist nur für Kopplungen zu einem anderen Rechner bzw. zu einer anderen Software brauchbar. Ein KISS-TNC lässt sich damit nicht vernünftig ansteuern!
  Bei der Verbindung mit einem RMNC oder PC/Flexnet-Komponenten sollte unbedingt eine Flexnet-kompatible CRC-Prüfsumme verwendet werden. Diese wird durch ein "c" im mode-Befehl eingeschaltet. Auch erfolgt die Einschaltung automatisch, sobald ein Paket mit Prüfsumme erkannt wird. Die Parametrierung des KISS-Interfaces erfolgt in *init.l2* mit:

  ```
  assign kiss       ; Wir wollen eine KISS-Verbindung
  device /dev/ttyS0 ; über die erste serielle Schnittstelle
  mode 19200c       ; mit 19200 Baud und Flexnet-CRC
  ```

- Zum anderen wird das AXIP-Protokoll mit UDP (nicht mit Raw-IP) unterstützt. Mit dessen Hilfe ist es möglich, zu anderen Rechnern über TCP/IP im AX.25-Protokoll zu kommunizieren. Dazu ist folgendes in *init.l2* einzutragen:

  ```
  assign axip       ; Wir wollen eine AXIP-Verbindung
  device 44.130.1.1 ; zum Rechner mit der Adresse 44.130.1.1
  port 4866         ; auf dem Port 4866
  ```

Zusätzlich kann bei Bedarf ein Parameter "txport <portnummer>" angegeben werden. Zu diesem hin erfolgen die Aussendungen, empfangen wird stets auf dem mit "port" angegebenem Port. Dies ist jedoch ein Spezialfall und normalerweise nicht üblich.

Gedacht ist diese Anschlussmöglichkeit in erster Linie zur Verbindung mit einem PC/Flexnet-Knotenrechner. Bei diesem muss *ippd.exe* von HB9JNX benutzt werden. Es wird dabei die Option UDP verwendet, also der Parameter "/u<port>". Die IP-Adresse der Gegenstelle wird in *init.l2* unter "device" angegeben. Dabei wird ein Name angegeben, der in */etc/hosts* einer IP-Nummer zugeordnet ist, oder direkt die IP-Nummer der Gegenstation. Es sollte ein Port über 1024 benutzt werden, da hierfür auf Mailboxseite keine root-Rechte erforderlich sind. Vorgeschlagen wird auch hier der Port 4866 wie bei TCP. *ippd.exe* wird dann so aufgerufen:

```
IPPD -i:<packetintno> [-c:<numchannels>] -m:<myipaddr>
     [-u:<port>] -p:<peeripaddr> [-g:<gwipaddr>[,<gwhwaddr>]] [-r:<proto>]
```

also z.B.:

```
IPPD -I:0x60 -M:44.130.1.1 -P:44.130.1.2 -U:4866
```

Die hier verwendeten IP-Adressen sind frei erfunden und sollten koordiniert sein, falls der Rechner in irgendeinem Netz eingebunden ist. Es ist darüber hinaus sinnvoll, die Hardwareadressen (gwhwaddr=Ethernetadresse) statisch einzutragen. In einem privaten Netz sollten Internetadressen laut RFC1918 verwendet werden: 10/8, 172.16/12, 192.168/16;

Es können mehrere Instanzen von AXIP oder KISS konfiguriert werden (insgesamt max. 20). Die Kanalzuordnung erfolgt in der Reihenfolge, in der die assign-Befehle in *init.l2* auftauchen. Wird nur eine Schnittstelle (also KISS oder AXIP) verwendet, so sollte diese sinnvollerweise als erster Eintrag in *init.l2* stehen. Wird KISS ohne CRC-Prüfung betrieben, so erfolgt ein #S-Eintrag ins Syslog. KISS sollte nur mit CRC betrieben werden, deshalb immer "c" beim Modebefehl in *init.l2* eintragen.

6.2.2. Radio connection using Linux with Kernel-AX.25

Die Konfiguration des Linux-Kernels wird hier nicht beschrieben. Bevor man sich daran versucht, die OpenBCM-Mailbox über das Kernel-AX.25 an die Außenwelt anzubinden, sollte man bereits mit dem Programm "call" aus den AX25-Utils erfolgreich versucht haben, einen Connect aufzubauen. Es sei hier auf das AX25- und das HAM-HOWTO verwiesen. Beides liegt normalerweise jeder Linux-Distribution bei.

Die ax25-utils sind nicht unbedingt notwendig, um die OpenBCM-Mailbox mit Kernel-AX.25 zu betreiben, wohl aber, um manche AX.25-Interfaces zu aktivieren (z.B. kissattach für KISS-Interfaces oder bpqparms für BPQ-Interfaces), außerdem empfiehlt sich das Programm *call* zum Testen der Konfiguration.

Hat man nun ein funktionierendes AX.25-Interface, gilt es zunächst herauszufinden, wie die Hardware-Adresse davon lautet. Angenommen das Interface heißt scc0, so lässt sich das mit dem Befehl "ifconfig scc0" bewerkstelligen. Die Ausgabe sieht z.B. so aus:

```
scc0      Link encap:AMPR AX.25  HWaddr DH3MB-10
          inet addr:44.130.186.1  Bcast:44.255.255.255  Mask:255.0.0.0
          UP RUNNING  MTU:256  Metric:1
          RX packets:10625 errors:1486940 dropped:0 overruns:0
          TX packets:2550 errors:0 dropped:0 overruns:0
```

Die Hardware-Adresse ("HWaddr") lautet hier DH3MB-10. Damit die OpenBCM-Mailbox nun über dieses Interface connected werden kann, ist nur ein einziger Befehl einzugeben:

```
ax25k_if dh3mb-10
```

Danach ist ein Neustart der Mailbox notwendig, um das Interface zum Kernel-AX.25 zu aktivieren. Zu beachten ist dabei, dass das Rufzeichen "dh3mb-10" nichts mit den Rufzeichen der Mailbox zu tun hat. Es ist lediglich eine Art Bezeichnung für das AX.25-Interface und wird von der Mailbox nicht weiter verwendet. Nun ist die Mailbox bereits über dieses Interface erreichbar. Um auch bei ausgehenden Connects (beim Sysop-Befehl CONNECT und bei S&F-Connects) das Kernel-AX.25-Interface zu verwenden, gibt es zwei Möglichkeiten:

- Man fügt zu obigem Befehl den Parameter "d" (für Default) hinzu, also z.B. "ax25k_if dh3mb-10 d", es wird dann bei allen ausgehenden Connects das Kernel-AX.25-Interface verwendet, d.h. ein Connect über den internen L2 ist nicht mehr möglich.
- Man gibt als ersten Parameter beim CONNECT-Befehl bzw. als drittes Feld in *fwd.bcm* einen String in der Form "ax:Hardware-Adresse" an, wobei die Hardware-Adresse des zu verwendenden Interfaces anzugeben ist. Der Befehl "CONNECT ax:dh3mb-10 DB0AAB DB0IRS" baut auf dem Kernel-AX.25-Interface mit der Hardware-Adresse DH3MB-10 einen Connect zu DB0AAB via DB0IRS auf. Innerhalb der Datei *fwd.bcm* baut die Mailbox, bei einem ausgehenden S&F-Connect zu DB0IRS, zuerst auf dem Kernel-AX.25-Interface mit der Hardware-Adresse DB0AAB-10 eine Verbindung zu DB0AAB auf, und connected dann weiter zur Mailbox DB0IRS-8.
  Wird dieser Parameter bzw. dieses Feld weggelassen, so erfolgt der Connect über den internen L2. Auch hier hat das Rufzeichen hinter "ax:" nichts mit dem eigenen Mailbox-Rufzeichen oder dem des Partners zu tun.

Abgeschaltet wird das AX.25-Interface mit "ax25k_if off".

6.3. Radio connection using Windows

Unter Windows NT/2000/XP ist die verfügbare Packet Radio-Software eher dünn gesät. Hier sei die Knotensoftware XNET erwähnt, die sich momentan problemlos über AXUDP mit der OpenBCM rechnerintern verbinden (bei unterschiedlichen RX/TX-Portnummern) lässt.

Des Weiteren ist es möglich mit Flex32 und AXIP-UDP mit der OpenBCM-Mailbox zu kommunizieren. Hier kann man dann auch z.B. ein Terminalprogramm wie Paxon (http://www.paxon.de) oder Winstop (http://www.winstop.de), die ebenfalls Flex32 unterstützen, direkt auf dem Mailbox-PC installieren. Auch eine Kombination OpenBCM, XNET, Flex32 und Terminalprogramm ist denkbar.

Die Fernbedienbarkeit des Betriebssystems Windows ist jedoch recht eingeschränkt.

Wird ein rechnerexternes Netzknoten-System verwendet, so erfolgt die Verbindung über serielle Schnittstelle mit KISS-Protokoll. Für bessere Performance ist jedoch eine Verbindung über Ethernet zum Netzknoten-System zu bevorzugen, sofern das Netzknotensystem dies unterstützt.

6.3.1. Configuration of OpenBCM using Windows

Bei der Windows-Version erfolgt der Packet-Radio-Anschluß wie auch bei der Linux-Version über *init.l2*.

Bei einer KISS-Verbindung über die serielle Schnittstelle wird hierbei allerdings nicht etwa /dev/ttyS0, sondern die unter DOS/Windows üblichen Bezeichnungen für die seriellen Schnittstellen angegeben, also COM1, COM2, etc. Das sieht dann in etwa so aus:

```
assign kiss        ; Wir wollen eine KISS-Verbindung
device COM1        ; über die erste serielle Schnittstelle
mode 19200c        ; mit 19200 Baud und Flexnet-CRC
```

Zum anderen wird, wie unter Linux auch, das AXIP-Protokoll mit UDP (nicht mit Raw-IP) unterstützt. Mit dessen Hilfe ist es möglich, zu anderen Rechnern über TCP/IP im AX.25-Protokoll zu kommunizieren. Dazu ist folgendes in *init.l2* einzutragen:

```
assign axip       ; Wir wollen eine AXIP-Verbindung
device 44.130.1.1 ; zum Rechner mit der Adresse 44.130.1.1
port   4866       ; auf dem Port 4866
```

6.3.2. Configuration of OpenBCM using Flexnet32 with Windows

Wenn man Flexnet32 und OpenBCM auf dem gleichen Rechner betreiben will, ist folgende Konfiguration der *init.l2* sinnvoll:

```
assign axip
peer   localhost
port   4722
txport 4721
```

Bei Flexnet32 muss dann im Control-Center zusätzlich ein AXIP-UDP-Port erstellt werden, der dann mit Port 4721 empfängt und auf Port 4722 sendet.

6.3.3. Configuration of OpenBCM using XNET with Windows

Will man XNET(NT) und OpenBCM auf dem gleichen Rechner betreiben, so ist in *init.l2* folgender Eintrag sinnvoll:

```
assign axip
peer   localhost
port   4724
txport 4723
```

Bei XNET muss in der *autoexec.net* ein AXIP-UDP-Port hinzugefügt werden, der dann mit Port 4723 empfängt und mit Port 4724 sendet. Dies ist beispielsweise ab Version 1.30 von XNET mit folgender Zeile machbar:

```
att ip0 axudp 0 1 l4723 d4724 127.0.0.1
```

6.4. Maximum amount of simultanous connections

Bei den meisten Mailboxsystemen (z.B. FBB, DieBox, W0RLI, MSYS) gibt es eine Grenze für die Anzahl der gleichzeitig in der Mailbox eingeloggter Stationen. Es sind dann z.B. nur 27 gleichzeitige Logins möglich.

Bei der OpenBCM-Mailbox wird für jedes Login dynamisch Speicher reserviert (ca. 5 kB unter DOS, ca. 20 kB unter Linux/Windows). Je nach Konfiguration (freier Speicher) ergibt sich eine praktische Grenze, die bei der DOS-Version bei etwa 80 gleichzeitigen Logins liegt. Unter 20 kB freien Speicher werden automatisch keine weiteren Logins zugelassen. Statisch wird die Anzahl der Logins bei DOS auf 100 Logins, bei Linux/Windows auf 200 Logins begrenzt.

Da damit eine eingeloggte Station anderen Stationen "nichts wegnimmt", gibt es bei der OpenBCM-Mailbox auch keine Reservierung bestimmter Kanäle als Sysop- bzw. Forwardkanäle: Es gibt eine ausreichende Anzahl von Kanälen, daher ist das nicht notwendig. Ein User-Timeout ist aus Sicht der Mailbox unnötig: Wer in der Box eingeloggt ist, kann schlafen so lange er will, sofern der Sysop ein Timeout nicht aus anderen Gründen aktiviert.

7. User interface

7.1. DOS interface

Der Monitor, der an den Mailbox-Rechner angeschlossen ist, ist die Systemkonsole. Die Mailbox stellt hier einen Bildschirm dar, der in drei Fenster unterteilt ist. Fast jedes Kommando zur Steuerung der Mailbox kann auch per Funk gegeben werden.

Die drei Fenster auf dem Bildschirm der Konsole sind:

- Monitor: Nach dem Start ist dieses Fenster leer. Mit "TNC TRACE -1" kann der Monitor nach dem Login eingeschaltet werden, dann wird hier der gesamte Packet-Betrieb mitgeschrieben. Wer diesen Befehl nicht bei jedem Neustart eingeben will, kann ihn auch in der Datei *startup.imp* eintragen.
- Console: Hier werden alle Systemmeldungen ausgegeben. Außerdem kommen alle Eingaben der Benutzer (nur Befehle) in diesem Fenster. Jede Meldung beginnt mit einem Buchstaben, der anzeigt, um welche Art Meldung es sich handelt:
  - Betriebsmeldungen (R=Report) werden nur am Bildschirm dargestellt, oder bei entsprechender Einstellung auch in die Datei *trace/syslog_r.bcm* geschrieben.
  - Fehlermeldungen (L=Log, S=Serious, F=Fatal, A=Abort) werden zusätzlich in die Datei *trace/syslog.bcm* geschrieben.

  Fehler der Kategorie S, F und A deuten entweder auf Softwarefehler hin, oder es sind schwerwiegende Fehler im Dateisystem. Auch grob fehlerhafte Sysopeingaben können zu Fehlermeldungen führen. Die Zahl der möglichen Meldungen ist sehr groß. Es wäre deshalb sehr mühsam, alle hier aufzuschreiben. Suspekte Meldungen bitte an BAYBOX @ BAYCOM schicken. Normalerweise findet sich jemand, der die Fehlermeldung deuten kann.
- Sysop: In diesem Fenster kann sich der Bediener in die Box einloggen. Beim Start erscheint ein Login-Prompt. An diesem kann entweder das eigene Rufzeichen angegeben werden (Login unter beliebigem Call) oder mit der Angabe von "." das in *init.bcm* eingestellte Sysop-Rufzeichen verwendet werden.

Links oben (über dem Monitor) wird die Task-Aktivität angezeigt. Jedes Zeichen entspricht einer Task in der Mailbox. Angezeigt werden folgende Zustände:

- I: Idle, also keinerlei Aktivität
- N: Neue Task, vor der ersten Suspendierung
- R: Task läuft gerade
- M: Bildschirmfenster wird aufbereitet
- U: Gesamtbildschirm wird neu eingerichtet

Rechts neben den Tasks wird der freie Speicher in Kilobytes angegeben.

Die drei Fenster auf der Konsole können mit folgenden Funktionstasten verwaltet werden:

- F1: Console-Fenster aktivieren
- F2: Monitor-Fenster aktivieren
- F3: Sysop-Fenster aktivieren
- F4-F10: Mailbox-Session 2..8
- CTRL-F3: Neues Box-Login Fenster (oder Befehl W2); Parameter: eigenes Rufzeichen; als zweiter Parameter kann das gewünschte MyCall der Box angegeben werden
- CTRL-F4: Editorfenster öffnen (oder Befehl ED <Dateiname>) In neueren Versionen der Mailbox ist der Editor nicht mehr enthalten (Speichermangel)
- CTRL-F5: (Zoom - ALT-Z) Fenster auf volle Bildschirmgröße setzen
- CTRL-F6: (Next - ALT-N) nächstes Fenster aktivieren
- CTRL-F7: (ALT-R) Fenstergröße ändern. Cursor: Move, SHIFT-Cursor: Resize
- ALT-F3: Fenster schließen (Task Kill, Vorsicht: kein Abspeichern) Jedes Fenster kann geschlossen, aber nicht unbedingt geöffnet werden (z.B. Console oder Monitor). Daher nicht unbekümmert anwenden.
- ALT-F5: DOS-Bildschirm zeigen (nur darstellen, Aussteigen mit OSHELL)
- ALT-C: Einstellen der Bildschirmfarben

Das Beenden der Software erfolgt mit ALT-X. Dabei werden alle eingeloggten Benutzer disconnected, alle Logins sauber getrennt und alle Dateien geschlossen. Sollte das nichts nutzen, so kann mit ALT-K ein harter Abbruch erfolgen, der allerdings nichts abschließt und somit offene Dinge hinterlassen kann. Normalerweise sollte jedoch auch dann nichts passieren, da das Dateisystem zu fast jeder Zeit konsistent ist.

Es ist sinnvoll, rechtzeitig vor dem Abstellen der Mailbox die Benutzer mit WALL <text> auf das Abstellen aufmerksam zu machen.

Mausbedienung: Ist am Rechner eine Maus angeschlossen, so können die Fenster auch mit der Maus bedient werden. Dabei gibt es folgende Möglichkeiten:

- Fenster anklicken: Fenster wird aktiviert kommt in den Vordergrund
- Fenster am oberen Balken anklicken und ziehen: Fenster verschieben
- Fenster rechts unten anklicken und ziehen: Größe verändern

7.2. Linux interface

Unter Linux wird nur das Syslog ausgegeben. Ein direkter Login ist nicht möglich. Dazu kann Telnet oder das Programm *bct* verwendet werden. Auf dem normalen Bildschirm kommt die Ausgabe der Trace-Meldungen (wie am "Console"-Fenster) und des Monitors, wie auf dem Monitor-Fenster. Das Abschalten des Monitors erfolgt mit "monitor off" in *init.l2*. Zur Kommunikation ist ein Web-Browser (z.B. Netscape) empfehlenswert.

Sinnvoll ist es, diesen Bildschirm auf /dev/tty8 auszugeben, dann kann man mit ALT-F8 den Boxbildschirm anwählen, sofern auf diesem kein getty aktiv ist. Die Zugriffsrechte müssen allerdings richtig eingestellt sein, damit eine Ausgabe darauf erfolgen kann.

Die Ausgabe im Trace-Fenster erfolgt gefiltert, nicht zulässige Zeichen werden durch einen Punkt ersetzt. Damit wird verhindert, dass Zeichen durch VT100-Terminals falsch interpretiert werden. Die Zeichen 0x0d 0x0c 0x20-0xf7 werden angezeigt, andere Zeichen werden durch einen Punkt ersetzt.

7.3. Windows interface

Die Windows-Version hat derzeit keine nennenswerte Benutzeroberfläche. Es öffnet sich ein Fenster, auf dem die eingestellten Trace-Ausgaben erscheinen. Der Zeichensatz passt sich so gut wie möglich der Fenstergröße an. Weitere Schnörkel sind nicht vorhanden.

Wie bei der Linux-Version wird über ein externes Programm auf die Box zugegriffen. Ein Login ist über *telnet.exe* möglich, das bei Windows NT/2000/XP mitgeliefert wird. Hier wird bei "Hostname" der eigene Rechnername und bei "Anschluss" der Port 4719 angegeben. Nach Verbindungsaufbau kommt ein Loginprompt wie in der DOS-Version.

8. General notes for sysop maintenance

The filesystem of the OpenBCM mailbox system is designed to be save for system crashes, wrong operating, deletion and falsification of files. This is the reason why you have only a few things remember in mind.

- Date/Time: Because the mail filenames are generated from date/timestamp and they must be unambiguous it is very important that the clock in your PC is running correct. Trustless or wrong time can lead to big troubles,

especially with the CHECK list file *check.bcm* (A binary search procedure will not work correct if the list is not correct sorted). If the clock is switch back while the mailbox is running, the internal clock of OpenBCM will run slower until the selected time is reached. During this time the mailbox should not be switched off. If this is necessary, in the switched back periode of time no new mail are allowed to be saved. If the time was running completly wrong you can reorganize your data with REORG F. But it is important that you start the REORG process after the doubled time periode, this means no mail file with a newer timestamp than the current time will arrive.

- Editing system files: When you manually edit system files *check.bcm* and *list.bcm* the mailbox system can be distrub extremly. If something happen with that files you should use the sysop command REORG F immediatly. This command repairs all *list.bcm* files and the file *check.bcm.*

- The machine readable files *users4.bcm* and *bids2.bcm* (or *bids3.bcm* in the Linux version) are extremly important for the mailbox. If they are damaged or deleted, the data has definitly gone away and so those files should be backuped in a constant period of time (it's recommended to create an automatic process for this)!

9. Complete forward configuration

The forward configuration is defined in file *fwd.bcm* which includes all needed information. The file discribes which forward partners exist and at which time which mails are send.

The reception of mails from the forward partners can't be influenced by *fwd.bcm*!

In file *fwd.bcm* is a section for each forward partner, which contains:

- the name of the forward partner (callsign without SSID)
- timetable, when forward connects should be made
- how to connect the forward partner (connection path)
- options for that forward partner section (e. g. maximum size of a mail, etc.)
- defintions which mails should be send to the forward partner

While the first points are easy to define, the last point is the one, which is really hard to define and where the most mistakes are made.

For bulletin mails exists the directors (that part, that is normally used with the SEND command after the "@"). For usermails the hierarchical address of the target home mailbox of the receiver is used.

Example for a usermail:

```
DF3VI @ DB0HOM.#SAR.DEU.EU
        ^Address (hierarchical address of target mailbox)
^Target callsign
```

Example for a bulletin:

```
LINUX @ DL
        ^Address (Director)
^Receiverboard
```

These are the two main used criteria for mail routing. Additional criteria are mentioned later.

9.1. The theory of analysis a hierarchical address

Following is an example of a hierarchical address:

```
DB0AAB.#BAY.DEU.EU
                ^ Continent EU (stands for Europe)
            ^ Country (DEU means Germany/Deutschland)
        ^ Possibly further definition of a region (#BAY means Bayern)
^ Callsign of a mailbox
```

The designator for continents is two or four letters long. In Europe you should only use "EU".

- .AF, .AFRC: Africa
- .AS, .ASIA: Asia
- .AU, .AUST: Australia
- .EU, .EURO: Europe
- .NA, .NOAM: North America
- .OC, .OCEA: Oceania
- .SA, .SOAM: South America
- .CEAM : Middle America
- .MDLE : MiddleEast (e.g. Israel)
- .ANTR Antarctica (*)
- .AUNZ Australia/New Zealand (*)
- .CAFR Central Africa (*)
- .CARB Caribic (*)
- .EPAC Eastern Pacific (*)
- .INDI Indean Ocean incl. Indian subcontinent (*)
- .MDLT Mediterranean (*)
- .NAFR Northern Africa (*)
- .NPAC Northern Pacific (*)
- .SAFR Southern Africa (*)
- .SEAS South-East Asia (*)
- .SPAC Southern Pacific (*)
- .WPAC Western Pacific (*)

The marked (*) designators are inventions of TAPR organization, practically never used and are only listed to be complete.

The designator of the different countries is specified by ISO 3166. The designator is three letters long. For Europe the following designators exists:

Practically used:

- .AUT Austria
- .BEL Belgium
- .BGR Bulgaria
- .BIH Bosnia-Herzegowina
- .CHE Swiss
- .CZE Czechia
- .DEU Germany
- .DNK Denmark
- .ESP Spain
- .EST Estonia
- .FIN Finland
- .FRA France
- .GBR Great Britain
- .GIB Gibraltar
- .GRC Greece

- .HRV Croatia
- .HUN Hungary
- .IRL Irland
- .ITA Italy
- .LTU Lithuania
- .LUX Luxemburg
- .LVA Latvia
- .MKD Mazedonia
- .MLT Malta
- .NLD Netherlands
- .NOR Norway
- .POL Polonia
- .PRT Portugal
- .ROM Romania
- .RUS Russia
- .SVK Slovakia
- .SVN Slovenia
- .SWE Sweden
- .TUR Turkey
- .UKR Ukraine
- .YUG Yugoslavia

Short form (can be used for copy & paste to your *fwd.bcm*):

```
.AUT .BEL .BGR .BIH .CHE .CZE .DEU
.DNK .ESP .EST .FIN .FRA .GBR .GIB
.GRC .HRV .HUN .IRL .ITA .LTU .LUX
.LVA .MKD .MLT .NLD .NOR .POL .PRT
.ROM .RUS .SVK .SVN .SWE .TUR .UKR
.YUG
```

Additions country designators:

- .ALB Albania
- .AND Andorra
- .BLR Belarus
- .FRO Färöer Island
- .GRL Greenland
- .ISL Island
- .LIE Lichtenstein
- .MCO Monaco
- .MDA Moldawia
- .MSR Monserrat
- .SJM Svalbard/Jan Mayen Island
- .SMR San Marino
- .VAT Vatican

By the way, Svalbard are belongings of Norway in the north polar ocean... ;-)

Following designators are obsolete and should not be used any more:

- .SUN Soviet union
- .CSK Czechoslovakia
- .MAK and .MAC Macedonia got only .MKD
- .TCH Czechia got .CZE since 1996

The regional designators can be defined using up to four letters. The first letter is always a "#".

Regional designators in Austria:

- .#OE1 Wien

- .#OE2 Salzburg
- .#OE3 Niederösterreich
- .#OE4 Burgenland
- .#OE5 Oberösterreich
- .#OE6 Steiermark
- .#OE7 Tirol
- .#OE8 Kärnten
- .#OE9 Vorarlberg

Regional designators in Czechia:
- .#MOR Mähren
- .#BOH Böhmen

Regional designators in Germany:
- .#BAY BAYern
- .#BLN BerLiN
- .#BRB BRandenBurg
- .#BW Baden-Württemberg
- .#HB Hansestadt Bremen
- .#HES HESsen
- .#HH Hansestadt Hamburg
- .#MVP Mecklenburg-VorPommern
- .#NDS NieDerSachsen
- .#NRW NordRhein-Westfalen
- .#RPL Rheinland PfaLz
- .#SAA SAchsen-Anhalt
- .#SAR SAaRland
- .#SAX SACHen
- .#SLH SchLeswig-Holstein
- .#THR THüRingen

If you use designator you should always know, that the address is analysed from right to left (only exception: a correct callsign is known). This means, that adding a ".#OE7" (Region in Austria) in a german mailbox will result to nothing, because this region will be searched in an addres like "OE7XKJ.#OE7.AUT.EU" from the right. ".EU" may not be omitted, because it defines the own area. ".AUT" is outside this area and so this must be added. Therefor the correct entry for Tirol in this example is ".#OE7.AUT". Additionally you can define for another forward partner ".AUT". OpenBCM is searching always for the longest complaince. This means that all usermail for Tirol are send to the partner which has ".#OE7.AUT" defined, and all other usermails for the rest of austria are send to the partner which section has only ".AUT" defined.

One special case is the used address .EURO at the end of the hierachical address. These cases must be defined individually (may be send .HUN.EURO to one partner and .ITA.EURO to another partner mailbox) because the analysis of the continent is not working here. OpenBCM is changing internally ".EURO" at the end of the hierarchical address into ".EU" – so you don't need to consider this special case. When forwarding such a mail, of course the original address is send.

9.2. The practical definition of forward entries

9.2.1. Usermails (using hierarchical address)

In the whole forward file following definitions must always exist:

- all continents must be defined, but without your own continent (e.g. all, but not .EU).
- all countries of own continent must be defined, but without your own country (e.g. all european countries but not .DEU).
- all regions in your own country must be defined, but without your own region (e.g. all german regions but not .#BAY).
- all mailboxes in the own region must be defined, but without the own mailbox (e.g. all mailboxes in region Bayern, but not the own one).

These entries must be splitted to all forward partners. You can define same entries for more than one partner mailbox. If a mail has to be send to this target, the mail is send to that mailbox, which is reached first. For the next mailbox the mail is deleted in the forward queue.

Important: it's really not useful to add entries like "A* B* C* ...", because with this, also useless address like @TE0ST.NOWHERE.XYZ are forwarded which cause to problems in your partner mailboxes! So, take some time and create a clearly defined *fwd.bcm* for your mailbox!

9.2.2. Bulletin mails (no hierarchical address)

There exist different directors for bulletin mails (a director is that, what is used after the "@" in the SEND command). Most used are:

- WW: worldwide
- EU: spread only in europe
- DL: only to the german speaking area (includes not only Germany!)
- OE: only to austria
- CZ: only to Czechia
- THEBOX: Used for THEBOX mailbox systems
- AMSAT: Satellit data
- BAYCOM: all OpenBCM mailboxes (used for OpenBCM software distribution)
- $WP: Used for White-Page/WPROT information (Mybbs etc.)

Some mailboxes use also further directors (e.g. @ALL, @TOUS etc.). But in praxis this often means the same like @WW. So, why not using always @WW if everybody should use the mail? It's hard to understand why such a nonsens is made. When you use OpenBCM you can clear up with these masses of directors: simply add lines to your *convat.bcm* file like

```
ALL WW
ALLE WW
TOUS WW
TODOS WW
```

...and so on. When now a mail with a director like @ALL reaches your mailbox it will be forwarded like it has @WW. When forwarding such a mail, of course the original address is send. So you have to define only WW in your forward sections where you need it. This clears up your *fwd.bcm* enormous.

If you want only one/some board(s) to be forwarded to your partner mailbox, you can define simply the board name(s) to the section in *fwd.bcm*. For sure, it's possible to send bulletins to more than one mailbox.

9.2.3. All forward options in overview

In file *fwd.bcm* you can define additional options for each section of a partner mailbox. Following options are possible:

```
  -b<bytes>  maximum size of forward mails (e.g. > 10k)
  -d         (delayed) forward only at set times of crontab.bcm
  -e         send E/M files with 4 lines in 1 frame (e.g. DB0SAO)
  -f         send empty line before start forward (e.g. TCPIP/xNOS)
  -i         BoxBin mails are forwarded regardless of the capabilities of
             the receiving system (not useful in most cases)
  -k         no forward of autobin bulletins
  -l         no forward of autobin user mails
  -m         no forward of 7plus bulletins
  -n         no forward of 7plus user mails
  -o         entries are sorted by size (smallest at first)
  -p         suppress prompt after login (e.g. needed for DB0SAO)
  -r         send ONLY boards from fwd.bcm to forward partner
  -s         SID is send immediately of connected BBS
  -t         tracing active (if "fwdtrace 2" is set)
  -u         connection is closed immediately, without this parameter, forward
             is waiting for some time if new mails can be forwarded
```

Note: The option "-l" is only useful, if the forward partner is really not capable with such mails. This is may the case if the target mailbox uses an obsolete software system. The option "-b" is only useful if the forward partner can only handle ASCII data (from WA7MBL). Internally OpenBCM limits the size of one mail at the moment to 10 MBytes. Every option must be separated through spaces, if you use more than one, for a section.

9.2.4. Special criteria (only) for bulletins

Normally bulletins are forwarded because of their director (e.g. @DL or @WW). But it's also possible that only some special boards are forwarded or special boards never being forwarded.

9.2.4.1. Special boards should never be forwarded

For this behaviour you have to add each board, you never want to forward, with an asterix in front of it in file *fwd.bcm*. Note: the name, before using the *convert.bcm* function, is taken.

Example:

```
   *MEINUNG *HUMOR *E
```

...will never send mail from boards MEINUNG, HUMOR and E.

9.2.4.2. Only special boards should be forwarded

For this behaviour you have to add in your *fwd.bcm* section the option "-R" and additionally the board(s) with an axterix in front of it. Only those mails are being forwarded, whose directors are added. Note: the name, before using the *convert.bcm* function, is taken.

Example:

```
   -R *BAYBOX *SYSOP *F
```

...will only send mail from board BAYBOX, SYSOP and F.

9.2.5. Forward timeout

While being forwarded and data reception is longer idle than defined with parameter FWDTIMEOUT, the forward connection is being disconnected and an entry in file *trace/syslog_r.bcm* is being made. You should select the timeout value in dependence of the forward link speed. Sometime even a value of 30 minutes is too

less! You can also use option "-U" to avoid the forward idle mode, but it's not recommented, because often new mails arrive and the new connection setup does more traffic than the held link.

9.2.6. Special forward options

For doing forward with some mailbox systems some special characteristics must be considered. This is mostly the case when doing forward with xNOS systems. All options are listed shortly in chapter "9.2.3. All forward options in overview". Here are some important options discribed again.

- Some xNOS system need to get a empty line after link has been setup, because those system detect with that, that it is a simple AX.25 and not a TCPIP connection. You can activate this feature with option "–F".
- The option "-T" causes the same as "FWDTRACE 2": the complete forward flow is being logged into a file *trace/t_<callsign>.bcm*. If you got a problem with connecting, this could be a big help.
- For forward partners, which can only handle a small amount of simultaneous connects, the feature to do each time a new mail forward connect after one new mail has arrived, can be switched of. If you add the option "-D" the forward connect is only made at the times, that are defined in file *crontab.bcm.* Note: the performance of a quick mail exchange is very poor in this case, and it's really not recommended to use this option, while not very important reasons exist for this.

9.3. Structure of a forward file

First here is an example of a simple *fwd.bcm* file:

```
;---------------------------------------------------------------
; fwd.bcm example file
;---------------------------------------------------------------
; Forward file of OE3XSR.#OE3.AUT.EU
;---------------------------------------------------------------
;      0         1         2
;      012345678901234567890123
;---------------------------------------------------------------
DB0WGS AAAAAAAAAAPAAAAAAAAAPAA DB0WGS-8
; BBS from own region
 OE3XBS
; Regions from .AUT
 .#OE2 .#OE5 .#OE8 .#CAR .#OE9 .#TIR .#OE7
; Europe
 .BEL .BGR .BIH .CHE .DEU .DNK
 .ESP .EST .FIN .FRA .GBR .GIB
 .GRC .HRV .IRL .ITA .LTU .LUX
 .LVA .MKD .MLT .NLD .NOR .PRT
 .ROM .RUS .SVK .SVN .SWE .TUR
 .UKR .YUG
; Continents
 .AF .AFRC .AS .AU .AUST .CEAM .CARB .MDLE .NA
 .NOAM .OC .OCEA .SA .SOAM
; Bulletin directors
 WW EU DL BAY OE
 BAYCOM AMSAT THEBOX
; White Page Infos for WPROT
$WP
;---------------------------------------------------------------
OE1XAB - OE1XAB OE3XBR
 -T
 .#OE1 .#OE4 .#OE6
 .HUN
 DL OE OEOST
 BAYCOM AMSAT
;---------------------------------------------------------------
OE3XZR - OE3XZR-8
 -T
 OE3XZR
```

```
;-------------------------------------------------------------
OK0NKT - OE3XNR / OK0NKT-12
 .CZE .POL
 WW EU AMSAT OK THEBOX
 $WP
;-------------------------------------------------------------
; END
;-------------------------------------------------------------
```

The first line of a forward section defines the partner mailbox name (callsign without SSID), the forward timing (when do forward which mails) and the connection path to the forward partner. After that, you can add as many lines as you want to define the usermail and bulletin behaviour. All these additional lines must have one space as first character in a line.

If no special case is need, you can define the forward timing with "-". This is equivalent with timing of "PAAAAAAAAAAAAAAAAAAAAA" and is best for most cases. All new received mails are immediately forwarded to the other partner mailboxes. Alternative you can define the timetable with a 24 character long string. Each character defines if mails are being forward in that hour. The first character symbolize the time from 0:00-0:59 o'clock, the second for 1:00-1:59 o'clock and so on. Following characters can be used:

- "A" (All): All mails (usermail and bulletins) are forwarded.
- "U" (User): Only usermail are forwarded.
- ".": Nothing is forwarded.
- "P" (Poll): All mails are forwarded (like "A"), and additionally at the time defined in *crontab.bcm* (normally each 30 minutes) a forward connect is being made to the forward partner, also if no mails are in the forward queue to send. This makes sense, if the partner mailbox has misconfigured the connect path and therefore can't send its mails to your BBS. It makes sense to do such a pool one time a day.

It's normally ok to add only "<target callsign> <node callsign>" for the connect path. In this case a connect is setup to <target callsign> via <node callsign>.

Example for a section in *fwd.bcm* (if a forward connection is setup, a connect to DB0IGL-8 via DB0IRS is made):

```
DB0IGL - DB0IGL-8 DB0IRS
 <entry1> <entry2> ...
 [...]
```

In some cases it's needed to make a multi-stage setup. In this case, you need to add between each stage a " / " in the connection path (take care, that you have to add a blank in front and behind that "/"!). Note: If you are using a flexnet node, a multi-stage link setup is not recommended and not necessary!

Example for a mulit-stage setup in *fwd.bcm*:

```
DB0WGS - DB0AAB / DB0HOB / DB0WGS-8 DB0WGS
 <entry1> <entry2> ...
 [...]
```

...this will lead in the following link setup when a forward connection is being made:

- Link setup to DB0AAB via AX.25 interface of OpenBCM (internal use of L2, PC/Flexnet or linux kernel-AX.25)
- **C DB0HOB**
- Wait for "connected to"
- **C DB0WGS-8 DB0WGS**
- Wait for "connected to"
- Now doing the forward

The string between the slashes is send transparent after "C" (this is the CONNECT command), therefore an entry like

```
    OE7XKJ - OE7XAR 3 / AX25 OE7XKJ-4
```

is also possible. In the above example first a connect to OE7XAR on port 3 is made, then the string "C AX25 OE7XKJ-4" is send. The syntax of connect path is also used for the user forward.

You can also use for multi-stage connects a "=" as first character. With this, the string until the next " / " is send transparent and without adding a "C" in front of it.

Example:

F5ABC AAPAAAAAAAAAAAAAAAAAAAA OE3XSR / =XCON F5ABC 3 / =BBS

...will connect first OE3XSR. After that, the string "XCON F3ABC 3" is send. If this connection is made, the string "BBS" is send. The answers "connected" etc. must be noticed before the next connect command is send. If a correct answer is not recognized, please send me a log to add this in future releases of OpenBCM. Due to local languages the "connected to" messages of systems vary extreme.

If an additional "=" exists, two commands are send transparent and without a "C" in front of it and without waiting for an answer between. If you add more "=" more commands are send.

Example:

DB0AC AAPAAAAAAAAAAAAAAAAAAAA telnet:db0ab.dyndns.org:23 / =DB0MY =GEHEIM =C 8:DB0AC / =BBS

...will connect first via DB0AC via telnet DB0AB.DYNDNS.ORG using port 23. As login string "DB0MY" is send. After that the string "GEHEIM" is send without waiting for an answer and after that also the string "C 8:DB0AC" is send without waiting for an answer. If connection is made with DB0AC (means a "connected to DB0AC" was recognized), the string "BBS" is send.

After defined the first line of a forward section you can add as much entries like you want which define, which mails are send to that target (each following line must start with a blank). You can add mailbox callsigns (e.g. "DB0IRS"), parts of mailbox callsigns (e.g. "OE*"), directors (e.g. "DL") and hierarchical address designators. The last must always start with a point in front of it, e.g. ".#BAY" or ".AUT", because an entry like "AUT" would be interpreted as director (means @AUT). Further on you have to worry about:

- If you want to exchange White-Page information, you have to add "$WP".
- If you have ambiguous entries the one with the longest accordance is being used. Example: OE7* and OE9* is send to one partner mailbox. OE* can be now added to another partner and will mean, all OE* without OE7* and OE9* is send to this partner. You should reduce the usage of such wildcards in *fwd.bcm* and indeed it's recommended to abandon the usage of wildcards, because it can make a forward configuration extremly confusing.

9.3.1. Additional notes for *fwd.bcm*

- If you want only receive from a partner but not send any mail, you have to edit only the callsign in the first line of a section. The timetable and the connect path isn't needed in this case.
- The syntax of capital or small letters is irrelevant.
- A ";" marked the rest of that line as a comment.
- Inside a forward section you can add spaces, linefeed and comments as much as you like. You have only to care that each additional line starts with a space.
- If a mail can't be forward because it's too long (when using option "-B" in *fwd.bcm*) a warning mail is generated for the sender callsign.
- Bulletins, which have already passed the own mailbox, are rejected.
- File *fwd.bcm* is initialized while the OpenBCM is starting. If it's changed while OpenBCM is running if can be re-initialized using the

sysop command NEW. After initializing, a message is displayed which show a small statistic (how many forward partners etc.).

- You should check with command "p –sfn" if there still exists addresses which can't be reached by your mailbox
- Changing the file *fwd.bcm* via remote access:
  a) - Read out the current file with command RTEXT *fwd.bcm*
     - Edit the file at your home PC
     - Send the modified file back with command WTEXT *fwd.bcm*
       (Hint: make a backup of the original file before!)
     - send the command NEW, to re-initialize the new forward file
  b) If option DF3VI_FWD_EDIT was used while compiling the mailbox software, you can use the internal editor. Note: you may loss edited comments in *fwd.bcm* when using the internal editor!
  c) Use external runutils of DF3VI if using OpenBCM for DOS

9.3.2. Section without forward

You can add a special forward section when you use the callsign "DUMMY" for that section. The sense of this is to add here mailbox callsigns and directors who should never be forwarded, but which will not added to file *trace/unknown.bcm* or to a forward queue file.

You can add e.g. a section (for the mailbox system at DB0AAB) like:

```
[...]
DUMMY - DUMMY
 DB0ABC DB0XYZ
[...]
```

...this results in:

- all mail for DB0ABC and DB0XYZ stay without any comment in DB0AAB (in this case maybe useful, because both neighbour mailboxes are offline at the moment),
- "DUMMY" will never be shown in "STATUS FORWARD" overview,
- no forward queue file (e.g. *trace/u_dummy.bcm*) are created for a "DUMMY" section.

The timetable (in above example "-") and the connection callsign must be added due to syntax reasons, so that the mails are really routed to "DUMMY".

9.4. More than one mailbox with the same callsign

If you are using more than one mailbox with the same callsign, it will normally lead to a lot of problems. This chapter describes what you have to do, to solve most of the problems. Nevertheless, it is really recommended not use more than one mailbox with the same callsign!

If you use more than one mailbox with the same callsign, the is always the wish, to exchange mails between these mailboxes. Such a mailbox configuration is not easy, and often users of one of your mailboxes have the problem to know, where their usermail can be found.

First it is important to learn about some terms:

- Callsign of a ham radio station (e.g. DB0PV or DL/OE3DZW/P)
- AX.25 address of a mailbox (e.g. DB0PV-8)
- Name of a mailbox (e.g. DB0PV)
- Complete hierarchical address of a mailbox (e.g. DB0PV.#BAY.DEU.EU); the name of the mailbox is the beginning of the hierarchical address.

If "more than one mailbox with the same callsign" is used, it means

- the AX.25 addresses of the mailboxes differ only with the SSID (e.g. DB0PV-8 and DB0PV-12).
- the names of the mailboxes are the same.
- the complete hierarchical addresses of the mailboxes are the same (!!) - this means, from outside you see only ONE address. (Side effect: The R:-lines are changed according to the sender mailbox, this means, it is not possible to see from outside which system is qrv, besides this can lead to more than one R:-line with the same hierarchical address in mails).
- each of your own mailboxes is only allowed to forward to one of your others own mailboxes.

If such a configuration will function, following conditions must complyed:

- send forward to a mailbox with the same name
- receive forward from a mailbox with the same name and mails with more than one same R:-line, especially those mails, where the own hierarchical address is shown in the R:-line.
- generete diffent BIDs between the mailboxes with the same callsign.
- Possibility needed, to forward usermails to another mailbox although the condition "MyBBS = mailbox address" was complyed.
- when using more than 2 mailboxen, forward only in circuit, no crossed network.

To fulfil the last point, a user can add a SSID to his MYBBS setting. This SSID is no part of a hierarchical address. It shows only, that the user does not have his real MyBBS in this mailbox but in another with the same mailbox callsign. In the forward file *fwd.bcm* the sysop has to add an entry like e.g. "DB0PV-12". If no SSID is used in MyBBS and the condition "MyBBS = mailbox address" is complyed the usermail is not forwarded. You can use a SSID for the MyBBS only, if you are using the option -L (means local) with command ALTER FORWARD.

Example:
At the station DB0PV two diffent mailboxes should be used with the callsign DB0PV. DB0PV-8 is one OpenBCM mailbox, and it should forward with another OpenBCM mailbox called DB0PV-12.

The entry in *fwd.bcm* at the mailbox DB0PV-8 will look like:

```
DB0PV - DB0PV-12
 DB0PV-12
```

In file *init.bcm* "boxaddress db0pv.#bay.deu.eu" and "mycall db0pv db0pv-8" will be defined.

The file *fwd.bcm* of second mailbox DB0PV-12 looks like:

```
DB0PV - DB0PV-8
 DB0PV-8
```

In file *init.bcm* the same "boxaddress db0pv.#bay.deu.eu" and "mycall db0pv-12" is defined.

A user of DB0PV-8, who wants to find his usermail in DB0PV-12, should enter in DB0PV-8:

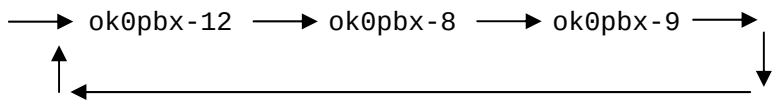
```
mybbs db0pv        (announce MyBBS DB0PV for the packet radio mailbox network)
mybbs -l db0pv-23  (forward locally to DB0PV-12)
```

A user of DB0PV-12, who wants to find his usermail in DB0PV-8 instead of mailbox DB0PV-12 should enter in DB0PV-12:

```
mybbs db0pv        (announce MyBBS DB0PV for the packet radio mailbox network)
mybbs -l db0pv-8   (forward locally to DB0PV-8)
```

If you are using more than one mailbox with the same callsign, you have to take care that your mailboxes will never create equal MIDs or BIDs. One possibility to make sure that this will never happen is to use diffent algorithm for creating MIDs/BIDs (e.g. F6FBB mailbox and OpenBCM mailbox). When using OpenBCM the sysop can't effect the generation of MIDs/BIDs.

Because the forward partner is only identified by name, it is important that those mailboxes only use a one way circuit for the network between each other, like for example:



If a mail should be send from OK0PBX-12 to OB0PBX-9, the mail must be first send from OK0PBX-12 to OK0PBX-8.

If you need more than one mailbox at the same place, check the possiblity to use each mailbox with a diffent callsign!

9.5. Initialization of file *fwd.bcm*

When initializing the file *fwd.bcm* each section for each forward partner is read in. Each entry is cut of the parts which contain in the own mailbox address.

Examples for the own mailbox DB0AAB.#BAY.DEU.EU:

a)

- DB0WGS.#BAY.DEU.EU
  .EU is contained in the own address, so it will be cut
- DB0WGS.#BAY.DEU
  .DEU is contained in the own address, so it will be cut
- DB0WGS.#BAY
  .#BAY is contained in the own address, so it will be cut
- DB0WGS
  remains, must be defined in *fwd.bcm*, because it's in the own region

b)

- K1XX.#BAY.NJ.USA.NA
  remains unmodified - .#BAY is namely a part of the own address, but because the analysis is started from right and doesn't match, it's unchanged

c)

- .#HES.DEU
  .DEU is cut, because it's contained in the own address
- .#HES
  remains

Now in the initialized data remain only entries, which lie outside the own area.

Besides, the callsign of the partner mailbox is added automatically to the section of the partner mailbox. This simplifies the analysis and also, an alternative routing is possible if the callsign should be added to another section.

9.6. Analysis of forward addresses

During the analysis of a forward address the first step is to look at the first field (means the callsign of the mailbox). If this is found exactly in the forward config file, the rest of the hierarchical address is ignored. Because

callsigns are clearly indicated, there is no imaginable case where this will lead into a wrong routing.

If the callsign itself is not found, the target forward address is cut (beginning from right to left) of that parts, that still are contained in the own mailbox address (see chapter "9.5. Initialization of file *fwd.bcm*").

After this the following procedure will be started: Each entry in *fwd.bcm* is compared with the end of the modified address. The longest complaince is searched. The condition for a match is a right-aligned exact complaince. Besides, the entry which is used for the search have to start with "." or is a valid mailbox callsign (because of address like DH5RAE.OE5XBL.AUT.EU).

If no possibilities match, the search is extended for wildcards. The wildcards must match the callsign of a mailbox (not a hierarchical address). Wildcards are only working with usermail addresses not with bulletin directors.

If no a match is not found, the mail is added to the logfile *trace/unknown.bcm* and will stay in your own mailbox. The logfile *trace/unknown.bcm* will not be machine read and should be re-viewed from time to time by the sysop. OpenBCM will generate a mail to the sender of a mail, that his mail will stay in the mailbox due to a routing error. In former time, a runutil UNK exist to generate such mail, but this tool is now obsolete.

If a match is found, the mail will be added in all matching forward queue files. You will find the queue files in subdirectory *fwd*, for each partner mailbox exist a file *u_<callsign>.bcm* for usermail queue and *i_<callsign>.bcm* for the bulletin queue. There is also a file *w_<callsign>.bcm* for WPROT message queue.

In the files for usermail and bulletins the boardname and the filename of the mail is added. To identify the board the precise boardname suffices, the complete path is not needed. The second field defines the size of the mail and the optional third field defines a timestamp (in UNIX format) which defines the earliest forward time for that mail.

Example of the syntax of one entry from a forward queue file:

YAESU/242FE73 <size> [<timestamp>]

If the mail has been forward, the first character in the mail queue file is overwriten with a blank. If the whole file contains only lines which first characters are blank, the complete file will be deleted.

The lines can vary in lenght and each line is finished with CR+LF.

If an entry still exist in the queue file, but the mail itself was deleted in the meanwhile, that entry is ignored and will be deleted from the mail queue.

If a mail should be forced to be forwarded to a partner mailbox (ignoring *fwd.bcm*) you can use the mailbox command FORWARD.

Before sending a mail the forwarding is check due to plausibility. Following errors can appear and are logged in SYSLOG:

- fwds: loop frombox>tobox
  The mail has run to often through your own mailbox (Loop) and stays now in your mailbox. Bulletins can only run one time through your mailbox, usermail up to three times. You can use commands FORWARD and MYBBS to overwrite this mechanism. The reason for a loop in bulletin forward is often that the BID has been changed in the meanwhile. The reason for a loop in usermail forward is most times a wrong forward configuration between two mailbox systems (each system is sending the mail to the other). This message can appear more than one time, because it will be

generated while sending (not instead of sending) the mail and will appear so for each partner mailbox where the mail has been tried to send to.

- fwds: too old frombox>tobox
  Happens only while forwarding bulletins which are older than the value defined with OLDESTFWD, the mail is too old and remains in your mailbox.
- fwds: size frombox>tobox
  The mailsize is bigger than the size defined by the "-B" option in *fwd.bcm*.

9.7. Forwarding of white page information

Userdata information is exchanged using the WPROT procedure of OE3DZW. If the neighbour mailbox can't handle this format (has no "W" in SID), the WP format from F6FBB mailbox system is being used automatically. You can find the specs of WPROT in chapter "30. Extended White Pages Protocol (WPROT) specification v1.0/Rev.2".

The obsolete E/M procedure - formerly established by THEBOX – can be handled too, but should not be used any more.

9.8. ACK messages

ACK messages are routed like usermails. The difference is, that

- they are only forwarded to partner mailboxes with an "A" in the SID.
- no R:-lines are added to such a received ACK message in forward.
- they are send with command "SA" in the forward.
- they are saved with internal flag "A".

The manual input of command "SA" at mailbox prompt is only possible, if you are logged in as sysop.

9.9. Starting forward

The forward connect to a partner mailbox is started if a new mail for that partner mailbox has been received and the partner mailbox could be reached at the last connect try or if the timetable in *crontab.bcm* defines a forward connect if the mailbox could last time not be reached. The sysop can also manually start a forward connect with "SF <callsign>" if needed. With "SF ALL" the forwarding for all partner mailboxes is started one by one.

After connection has been established, first, usermails are send, after that the bulletins and just before the end, the WPROT mails. WPROT mails are only forwarded if bulletins or usermail are in the forward queue. They will be only send alone, if the forward was started by *crontab.bcm*.

To start the forwarding each half hour, you have to add a line in *crontab.bcm* like following:

```
[...]
0,30       *     *   *     *       forward
[...]
```

9.10. The autorouter

There is the possibility to use the autorouter function of OpenBCM. To use it, you have to make sure, that you mailbox software was compilied with the option _AUTOFWD in *config.h* of source code.

The Autorouter only tries to find a routing, if no valid entry in *fwd.bcm* was found for that target. So, this means, you can use the autorouter as an addition for the normal configuration of *fwd.bcm*, entrys in *fwd.bcm* always overwrite possible autorouter targets.

You can activate the autorouter by setting the parameter AUTOFWDTIME in file *init.bcm*. With this parameter you define the amount of days, how long an entry of received mails can be used for calculating a routing. Normally a value of 30 days is useful. If you set the value to 0, the autorouter is switched off.

To get an overview which routing are calculated by autorouter, you can use the command AFWDLIST. As a result of this command you get a list of all found autoroutings for your partner mailboxes. More information is saved in file *afwd1.tmp* which can be found in subdirectory *temp* of your mailbox and which has following syntax:

<partner callsign> <target mailbox+header> <shorted header> <age of info>

The autorouter gets the best possible path while calculating the amount of hops and the transmission duration between the target and your mailbox. One additional condition is that at least 4 mails must be received over that path.

The command AFWDLIST creates also the file *afwd.bcm*, if AUTOFWDTIME is not set to 0. In this file you will find also the calculated additional routings. Because you don't want to enter this command each time manually you should add in your file *crontab.bcm* a line "afwdlist", so that this is executed automatically e.g. once a day (recommended value):

```
[...]
10        2    *    *     *      afwdlist
[...]
```

The forwarding of mail will work like follwoing if autorouter is activated:
1.) *fwd.bcm*: as usual, only if no match found go on to 2.)
2.) *afwd.bcm*: this file was generated by autorouter
3.) *hadr4.bcm*: Database is searched (makes only sense for new entries, otherwise they still exist in *afwd.bcm*, but this methode is so quick, that it can be used)

With command "AUTOPATH [-a] <callsign>" you can check, if a route for <callsign> is known by your mailbox.

9.11. Active routing

Since version 1.06 of OpenBCM the ACTIVE-ROUTING function is implemented. Don't confuse ACTIVE-ROUTING with the autorouter feature (see chapter "9.10. The autorouter"), which is analysing received mails and add forward routes as a result of that. ACTIVE-ROUTING takes a step forward: here the forward systems exchange automatically their information about their neighbours and reachable target mailboxes with delay times and target routing path. Only the current best path is being used for routing usermail to the target. If an OpenBCM system recognize that the neighbour can handle ACTIVE-ROUTING, usermails are forwarded to the known target via ACTIVE-ROUTING. Manually entries in *fwd.bcm* are overwritten for those targets. You can find the specs of WPROT and ACTIVE-ROUTING in chapter "30. Extended White Pages Protocol (WPROT) specification v1.0/Rev.2".

9.12. IGATE in connect path

In the last 2 years a kind of internet backbone system war born for the ham radio packet network. This system can be connected with the callsign "IGATE"

(=Internet GATEway). Some sysops with badly working HF equipment but with stable money and internet access began now to use IGATE also in the forwarding connect path. OpenBCM is now so designed, that a string "IGATE" in connect path is only working, if the last connect try was unsuccessful. With this behaviour, the forward will automatically try from time to time the connect via the (slower) HF path without the sysop have to change the path each day to changing conditions.

If in connect path of one partner mailbox of *fwd.bcm* the callsign "IGATE" was added, it is ignored, if
- it is the first connect try since initializing of *fwd.bcm*.
- the last connect to the partner mailbox was successful.

In praxis this results in, that the connect will be tried automatically from time to time via the HF links (without IGATE). If this doesn't work the next try is a connect via IGATE, to help the forward to stay alive, if HF is really working badly. It's clear that IGATE should be defined in *fwd.bcm* at the right place (if IGATE should not be used, simply don't add it).

Example:
```
HB9EAS AAAAAAAAAAAAAAAAAAAAAAPA DB0FHN / IGATE / HB9EAS-8
```
or
```
HB9EAS AAAAAAAAAAAAAAAAAAAAAAPA HB9EAS-8 DB0FHN IGATE
```

If IGATE is not used for the forwarding (means HF is working), you will get following output if you enter the command DIR PATH:
```
HB9EAS AAAAAAAAAAAAAAAAAAAAAAPA HB9EAS-8 DB0FHN IGATE
                currently used: HB9EAS-8 DB0FHN
```

The function of IGATE is irrelevant for citizen band packet radio.

9.13. Telnet forward using Linux or Windows

The Linux and the Windows version (not DOS version) include the possibility to use a telnet connection for doing the forward. To let OpenBCM know, that you would like to use a telnet connection, you have to define the telnet address instead of the connect path in *fwd.bcm*. The following example should show how the syntax looks like:
```
PWR48N AAAAAAAAAAAAAUUUUAAAAAAP telnet:pwr48n.dynu.com:23
```

If there is a new mail for the telnet partner mailbox, the forward will start making a telnet connect using the defined telnet address and port (you can also use an IP number instead of the address). If the connect can be established, your mailbox is sending automatically its own callsign for login purpose. If the other mailbox needs a password, you can define it with
```
SETUSER <callsign> PW <password>
```
while <callsign> is the callsign of your partner mailbox. So, if a password is defined for the partner mailbox in this way, this password is also send automatically after the login.

Note: if your partner mailbox doesn't send its SID after login, you can use the option -S in *fwd.bcm* to send your SID without waiting for the other.

9.14. File forward

If the source file option _FILEFWD was used, when your mailbox was compiled, there is the possibility to use file forward with other mailbox systems. The forward file is (like F6FBB system use) in ASCII format.

To import a forward file you have to enter the command
```
FWDIMPORT <callsign> <filename>
```
The mails are saved in your mailbox like they were received from <callsign>.

To export mails, you can use the following command:

FWDEXPORT [-<option>] <callsign> <filename>

In this case, all queued mails for <callsign> are exported to <filename>.
You can also use <option> to change the export behaviour:

```
without <option>   the SID "AFHMR$" is used (compatible with F6FBB)
-A                 BAYBOX extensions are used (Autobin, WPROT-Format)
-D                 TheBox extensiosn are used (Autobin, E/M-Mails)
-S<sid>            you can define your own SID for using file forward export
```

In file *fwd.bcm* you make a normal section entry for your partner mailbox. You can even use a connect path (for normal connect behaviour) and only manually export mails for your partner mailbox if e.g. the hf link is not working, but you can the mail on a cd or disc. If you want only use file forward, you can replace the connect path in *fwd.bcm* with

FILE[: <IMPORTFILENAME> <EXPORTFILENAME> [<EXPORTOPTION>]]

Example:

DB0ABC AAAAAAAAAAAAAAAAAAAAAAAP FILE: DB0ABC.IMP DB0ABC.EXP D

or

DB0ABC AAAAAAAAAAAAAAAAAAAAAAAP FILE

If there is a new mail for the telnet partner mailbox, they are automaically exported to file *bcm/fwd/export/<EXPORTFILENAME>*, and also new mails are imported from file *bcm/fwd/import/<IMPORTFILENAME>*, if the forward is starting (by timetable or manually with "SF <BOXCALLSIGN>"). You don't have to add the import or export filename (restriction: at the moment the filenames must be only in capital letters!) or possible export options. If you simply add "FILE" with anything else, the import filename will be *<BOXCALL>.imp* and the export filename will be *<BOXCALL>.exp* (here the suffix will be only in small letters, so that the linux tool *sftool* can be used with a patch).
The directorys *bcm/fwd/import* and *bcm/fwd/export* are automatically created if they don't exist. When using the command ST F you will see status "filefwd" for all partner mailboxes which has as the connect path "FILE".

And can you send your exported forward files to your partner mailbox?
When using Linux, you can try the tool "Sftool v0.5a" (have a look at http://www.wspse.de or ftp://excelsior.kullen.rwth-aachen.de/pub/packet_radio/dpbox/sftool-0.5a.tar.gz)
made by Matthias Hensler to send the files automatically. This tool was formerly developed for DPBox, but you can use it without problems also with OpenBCM.

When using Windows, you can try the freeware tool "Automailer v1.3.3" (have a look at http://www.duodata.de/automailer), which can send the exported forward files automatically.

10. Detection of 7plus mails

While receiving a mail, OpenBCM tries to detect 7plus coded parts. Inside the bulletin lists a prefix "(7+)" is added if a 7plus part is detected. When forwarding such a mail to a partner mailbox, it's sure, that the original title of the mail is used (without "(7+)").

There is also the possibility that the sysop can configure the BBS in such a manner that the mailbox will notify or even delete mails with defect 7plus parts inside. This can be configured with the parameter SAVEBROKEN in file *init.bcm*.
Defect mails are marked with "(7-)" in bulletin lists.

Following configurations are possible:

- SAVEBROKEN 0: Only correct 7plus mails are accepted, defect 7plus mails are not stored in forward.

- SAVEBROKEN 1: Doing FBB forward, all 7plus mails are accepted; doing an ASCII forward, only mails with correct line checksum and length are accepted, others are not stored
- SAVEBROKEN 2: All mails – even such with defect 7plus parts inside – are accepted in forward

Note: The parameter SAVEBROKEN has only influence on mails, which are received by forward. If a user is sending a defect 7plus mail, it's always rejected and not stored!

11. Password functions

To have the best possible compatiblity with other systems, OpenBCM is supporting the BayCom/TheNet and also the MD2/MD5 password procedure. The old DieBox password procedure is supported as well, but it is hardly suggested that this is not used any longer, because a lot of other systems have problems with it, after 31. december 1999 ("year 2000 bug").

11.1. BayCom password procedure

Das BayCom-Passwort-Verfahren, das dem bei TheNet ähnelt, basiert darauf, dass beide Seiten (User und Mailbox/Digi) über einen identischen String fast beliebiger Länge (üblicherweise nicht mehr als 80 Zeichen) verfügen. Bei der Passwort-Eingabe fragt die Mailbox bzw. der Digi nun nach fünf Zeichen aus diesem String. Der User sendet diese fünf Zeichen zurück. Falls die Zeichen korrekt waren, war auch die Passwort-Eingabe erfolgreich. Damit eventuelle Mitleser nicht bei jeder Passwort-Abfrage fünf Zeichen des Passworts herausfinden können, kann das Passwort in einem maximal 80 Zeichen langen Zufalls-String "versteckt" werden.

Je länger das Passwort ist, desto sicherer ist es. Die Software sollte außerdem verhindern, dass bei einer Passwortabfrage zweimal das gleiche Zeichen abgefragt wird, da sonst das Passwort relativ einfach im Zufalls-String erkannt werden kann.

11.1.1. Sysop identification using BayCom password procedure

Um mit dem BayCom-Passwort Sysop-Status zu erlangen, gibt man einfach nur

```
PW
```

ein. Die Box sendet dann eine Leerzeile gefolgt von einem String wie z.B.

```
DB0AAB> 23 67 19 47 2
```

zurück. Diese Zahlen entsprechen den Stellen im Passwort-String, die der User zurücksenden muss (siehe oben). Das Sysop-BayCom-Passwort wird in der ersten Zeile der Datei *passwd.bcm* eingestellt. Der Befehl SYS bewirkt das gleiche, es wird vor der Passwort-Abfrage jedoch keine Leerzeile ausgegeben.

11.1.2. User login using BayCom password procedure

Sofern der Sysop es mit dem Befehl "USERPW 1" erlaubt hat (es handelt sich dabei um eine Einstellung in *init.bcm*), kann sich jeder User selbst ein Passwort setzen, das dann vor dem Login abgefragt wird. Das BayCom-Passwort-Verfahren wird aktiviert durch die Eingabe von:

```
A(LTER) LO(GINPWTYPE) BAYCOM
```

Die Eingabe des Passworts selbst erfolgt durch

A(LTER) PW <Passwort-String>

Das Passwort kann dabei in mehreren Schritten eingegeben werden, d.h. bei der wiederholten Eingabe dieses Befehls wird das Passwort nicht jedes Mal überschrieben, sondern der String wird jeweils an das aktuelle Passwort angehängt. Die maximale Länge des User-Passworts beträgt 39 Zeichen.

Sind nun diese beiden Einstellungen vorgenommen, so erhält man nach dem Connect der Mailbox nicht wie gewohnt den Connect-Text, sondern eine Passwortabfrage wie z.B.:

DB0AAB> 9 19 27 12 38

Wie auch beim Sysop-Passwort muss man dann mit den entsprechenden Zeichen aus dem Passwort-String antworten, die wiederum in einem Zufallsstring versteckt sein dürfen. Sind die zurückgesendeten Zeichen falsch, so wird die Verbindung getrennt, ansonsten erfolgt der Login wie gewohnt.

Durch Eingabe von "A(LTER) PW OFF" oder auch "A(LTER) PW" ohne weitere Parameter wird das Passwort gelöscht und die Passwort-Abfrage vor dem Login deaktiviert. Möchte man die Passwort-Abfrage nur vorübergehend abschalten, ohne das Passwort selbst zu löschen, so ist das durch die Eingabe von

A(LTER) LO(GINPWTYPE) OFF

möglich. Das Login-Passwort lässt sich auch soweit abschalten, dass ein erneutes Einschalten nur noch durch den Sysop möglich ist. Dies geschieht durch die Eingabe von:

A(LTER) PW DISABLE

Bei älteren Terminal-Programmen ist es oft notwendig, dass das Programm selbst eine Zeile mit einem bestimmten Befehl schickt (z.B. *pw*), damit die Passwort-Abfrage automatisch beantwortet wird. Um auch zu solchen Programmen kompatibel zu sein, ist der Befehl "ALTER PWLINE 1" gedacht. Nach dessen Eingabe wartet die Mailbox vor dem nächsten Login auf eine Zeile mit beliebigem Inhalt, bevor die Passwort-Abfrage gestartet wird. Diese Verhaltensweise lässt sich mit "ALTER PWLINE 0" wieder abstellen.

11.1.3. Store&Forward using BayCom password procedure

Beim (User-)Store&Forward mit dem BayCom-Passwort-Verfahren wird wie beim User-Passwort der String verwendet, der mit "ALTER PW" eingestellt wurde. Ein User, der Passwort-geschützten User-Store&Forward betreiben will, wird den Passwort-String also wie beim Login-Passwort einstellen. Beim Forward mit einer anderen (offiziellen) Mailbox jedoch wird der Sysop das mit dem Befehl

SETU(SER) <Rufzeichen der anderen Mailbox> PW <Passwort-String>

tun. Das BayCom-Passwort-Verfahren beim (User-)Store&Forward wird aktiviert mit

A(LTER) SFPWTYPE BAYCOM

beim User-Store&Forward bzw.

SETU(SER) <Rufzeichen der anderen Mailbox> SFPWTYPE BAYCOM

beim Forward mit einer anderen offiziellen Mailbox.

Alle anderen Einstellungen bzgl. des Passworts können analog zum Login-Passwort vorgenommen worden. D.h. das Löschen das Passworts erfolgt mit "ALTER PW OFF" bzw. "SETUSER <Rufzeichen der anderen Mailbox> PW OFF" und eine vorübergehende Abschaltung des Passworts ohne Löschung des Passwort-Strings selbst erfolgt mit "ALTER FWDPWTYPE OFF" bzw. "SETUSER <Rufzeichen der anderen Mailbox> FWDPWTYPE OFF".

Ist das Passwort nun auf beiden Seiten eingestellt (es muss natürlich der gleiche String und das gleiche Verfahren verwendet werden!), und connected eine Mailbox den Forward-Partner, so schickt die connectete Box nicht wie gewöhnlich nur das SID, sondern das SID gefolgt von einer Passwort-Abfrage. Das sieht dann z.B. so aus:

[OpenBCM-1.05-AB1D1FHMRW$] 7 23 49 87 34

Bei den Zahlen handelt es sich wie beim Sysop-/User-Passwort um Stellen aus dem Passwort-String. Die andere Mailbox antwortet z.B. mit

```
[DP-5.07-AB1D1FHMR$]  j823t5cl453by9h091n9lq
```

also mit ihrem SID, gefolgt von einer Zeichenfolge, in der die abgefragten Zeichen des Passwort-Strings versteckt sind. Ist die Antwort korrekt, so geht es mit dem Forwarding wie gewohnt weiter, ansonsten wird eine Fehlermeldung ausgegeben, und die Verbindung getrennt.

11.2. MD2/MD5 password procedure

Beim MD2/MD5-Passwort-Verfahren wird nicht das Passwort selbst bzw. ein Teil des Passworts übertragen, sondern eine Zahl, die aus einem Zufallsstring und dem Passwort berechnet wird. Dieser Zufallsstring wird vom System, auf dem man sich identifizieren will (also z.B. der Mailbox) bestimmt. Nachdem dieser String dem anderen System mitgeteilt wurde, wird an ihn auf beiden Seiten das Passwort "angehängt". Aus der auf diese Weise entstandenen Zeichenkette wird nach dem MD2/MD5-Algorithmus (dokumentiert in RFC 1319 und RFC 1321) eine Zahl berechnet. Diese Zahl ist 128 Bit groß und wird zu dem System, auf dem die Passwort-Abfrage gestartet wurde als 32-stellige Hexadezimal-Zahl übertragen. Dort wird diese Zahl überprüft, und bei einer Übereinstimmung die Passwort-Eingabe als erfolgreich betrachtet. Das MD2-Passwort unterscheidet sich vom MD5-Passwort nur dadurch, dass der Algorithmus zur Berechnung der 128 Bit-Zahl verschieden ist, das MD5-Passwort ist dadurch minimal sicherer.

Zu beachten ist, dass der angesprochene Zufalls-String exakt 10 Zeichen lang sein muss, und nur aus Zahlen bestehen darf. Diese Einschränkung ist rein willkürlich und wurde durch die ersten Implementationen in verschiedenen Mailbox-Systemen festgelegt.

MD2 und MD5 sind übrigens ursprünglich keine Passwort-Verfahren, sondern Algorithmen zur Berechnung von CRCs, um den Inhalt zweier Dateien auf Gleichheit zu überprüfen. Das Ziel beider Verfahren ist dabei eine maximale Änderung der berechneten Zahl bei einem minimalen Unterschied innerhalb der beiden Dateien.

11.2.1. Sysop identification with MD2/MD5 password procedure

Um mit dem MD2/MD5-Passwort Sysop-Status zu erlangen, gibt man einfach nur

```
MD2 (Beim MD2-Passwortverfahren)
```

bzw.

```
MD5 (Beim MD5-Passwortverfahren)
```

ein. Die Box sendet dann eine Leerzeile gefolgt von einem String wie z.B.:

```
DB0AAB> [1234567890]
```

zurück. Die Zahlen innerhalb der eckigen Klammern sind der Zufallsstring, der an das Passwort angehängt wird. Daraus wird mit dem MD2- bzw. mit dem MD5-Algorithmus eine Zahl berechnet, die dann zur Mailbox geschickt wird (siehe oben). Das Sysop-MD2-Passwort wird in der zweiten Zeile in der Datei *passwd.bcm* eingestellt, das MD5-Passwort in der dritten.

11.2.2. User login with MD2/MD5 password procedure

Der Passwort-String wird wie beim BayCom-Passwort-Verfahren mit "ALTER PW" eingestellt. Die Aktivierung des MD2/MD5-Passwort-Verfahrens beim User-Login erfolgt mit

```
A(LTER) LO(GINPWTYPE) MD2
```

bzw.

```
A(LTER) LO(GINPWTYPE) MD5
```

Die Passwort-Abfrage und die Berechnung der Antwort erfolgt analog zur Sysop-Identifikation.

11.2.3. Store&Forward using MD2/MD5 password procedure

Der Passwort-String wird wie beim BayCom-Passwort-Verfahren mit "ALTER PW" eingestellt. Die Aktivierung des MD2/MD5-Passwort-Verfahrens beim (User-) Store&Forward erfolgt mit

A(LTER) FW(DPWTYPE) MD2 bzw. MD5 (beim User-Store&Forward)

bzw.

SETU(SER) <Rufzeichen der anderen Mailbox> FW(DPWTYPE) MD2 bzw. MD5

beim Forward mit einer anderen offiziellen Mailbox.

Die Passwort-Abfrage beim Store&Forward erfolgt analog zum BayCom-Verfahren, die Zufalls-Zahlenfolge wird in eckigen Klammern hinter dem SID mit einem Leerzeichen getrennt übertragen.

11.3. Changes after successful sysop identification

Nach der erfolgreichen Sysop-Passworteingabe stehen zusätzlich zu den User-Kommandos auch alle Sysop-Funktionen zur Verfügung. An den User-Kommandos ändert sich folgendes:

- Der Sysop ist Eigentümer von allen Mails, das bezieht sich auf die Befehle ERASE, UNERASE, SETLIFE, FORWARD und TRANSFER.
- Alle Eingaben des Sysops werden in der Datei *trace/cmdlog.bcm* gespeichert.
- Beim ERASE-Befehl kommt die Option -F hinzu. Wird diese Option angegeben, wird die Nachricht nicht nur lokal gelöscht, sondern es wird auch noch eine Remote-Erase-Mail erzeugt. Dies sollte allerdings nur bei wirklich offensichtlich AFu-schädlichen Mails angewendet werden, um den Ruf einer Zensur zu vermeiden.

Merke: Das persönliche Empfinden, dass eine Mail gelöscht werden soll, unterscheidet sich im Allgemeinen von einem Sysop zum nächsten. Dem sollte dringend Rechnung getragen werden und nur dann vom globalen Löschen Gebrauch gemacht werden, wenn keinerlei Zweifel besteht, dass die Nachricht in den Abfall soll.

- Boards mit einstelligen Namen wie F oder T können gelesen und beschrieben werden. Einstellige Boards haben teilweise eine besondere Bedeutung.
- Mit Sysop-Berechtigung sind auch gelöschte Nachrichten lesbar, ein UNERASE ist daher zum Lesen nicht erforderlich.

Bezüglich des Sysop-Status sind folgende Dinge zu beachten:

- An der DOS-Mailbox-Console ist man nach dem Login immer Sysop. Das gleiche gilt, wenn man sich bei einer Linux/Windows-OpenBCM-Mailbox von einem Rechner aus einloggt, dessen Hostname oder/und IP-Adresse in der Datei *rhosts.bcm* angegeben ist. Ferner kann man die Datei *asysop.bcm* zur Autosysop-Funktion nutzen.
- Will man den Sysop-Status wieder ablegen, so ist das durch die Eingabe des Befehls PW OFF möglich.
- Wenn die Box DISABLED ist, dann ist ein Einloggen nur durch die Eingabe des Passworts möglich.

11.4. Secure Store&Forward with password between different mailbox systems

Wird zwischen verschiedenen Mailbox-Systemen geforwardet, so ist die Verwendung des MD5-Passwort-Vefahrens sinnvoll.

11.5. Principles of the different password procedures

Da die Passworteingabe von jedermann mitgelesen werden kann, erfolgt nicht, wie bei anderen Systemen (z.B. UNIX) üblich, die Eingabe einer "geheimen" Zeichenkette, sondern es werden entweder nur Teile des Passworts abgefragt (BayCom-Verfahren), oder das Passwort wird vor der Übertragung auf eine bestimmte Weise "verschlüsselt", wobei diese "Verschlüsselung" natürlich bei jeder Passwort-Übertragung verschieden sein muss (MD2/MD5-Passwort-Verfahren). Dabei wird der String, mit dem das Passwort "verschlüsselt" werden soll, von der Mailbox bestimmt. Dieser String muss absolut zufällig sein, darf sich also keinesfalls innerhalb eines kurzen Zeitraums wiederholen und die Verschlüsselung darf sich auf keinen Fall umkehren lassen (sonst kann ja jeder das Passwort berechnen).

11.5.1. Principle MD2/MD5 password procedure

Wie oben schon beschrieben wird hier dem Passwort ein von der Mailbox erzeugter Zufalls-String vorangestellt und aus der entstandenen Zeichenkette eine 128 Bit-Zahl berechnet. Diese wird dann zur Mailbox übertragen, wo anschließend überprüft wird, ob auf beiden Seiten das gleiche Passwort vorliegt.

Da bei dieser Verfahrensweise nie das Passwort selbst oder ein Teil des Passworts übertragen wird, kann ein Mitleser auch nie einen Teil des Passworts herausfinden. Auch die 128-Bit-Zahl selbst, kann ein "Einbrecher" unmöglich erraten. Die einzige theoretische Möglichkeit wäre, mit einer mitgelesenen Kombination aus Zufallsstring und zur Mailbox gesendeten Zahl alle Passwort-Möglichkeiten durchzuprobieren, und so das Passwort zu "knacken". Da es aber bereits bei einem fünf-stelligen Passwort über eine Milliarde Kombinationsmöglichkeiten gibt, kann ein MD2/MD5-Passwort durchaus als "unknackbar" angesehen werden.

11.6. All password procedures for user and forward in overview

Alle Benutzer können sich selbst ein AX25-Passwort einstellen, sofern dies vom Sysop freigegeben wurde. Zum Login bei HTTP und POP3 via Amateurfunk ist als Passwort der eigene Vorname einzugeben. Beim Zugriff über Drahtnetze (im allgemeinen Internet) und beim CB-Funk ist ein vom Sysop einzustellendes TTY-Passwort notwendig.

Zugriff	Mögliche Passwortverfahren	Befehle
Allgemein:		
AX25	BayCom, MD2, MD5, Priv	a pw, a pwline, a loginpwtype, pw
AX25-S&F	BayCom, MD2, MD5, Priv	a pw, a sfpwtype
Sysop	BayCom, MD2, MD5, Priv	sysop, priv, md2, md5
nur DOS:		
Modem	TTY-Passwort	a ttypw (nur Sysop)
nur Linux und Windows:		
Telnet	TTY-Passwort	a ttypw (nur Sysop)
Telnet-S&F	BayCom, MD2, MD5, Priv	a pw, a sfpwtype und a ttypw
POP3-Afu	Vorname	a name
POP3-Internet	TTY-Passwort	a ttypw (nur Sysop)

SMTP-Afu	**Vorname**	**a name**
SMTP-Internet	**TTY-Passwort**	**a ttypw**
HTTP-Afu	**Vorname**	**a name**
HTTP-Internet	**TTY-Passwort**	**a ttypw (nur Sysop)**

PRIV wird verwendet, sobald eine Datei *<call>.bcm* bzw. *priv.bcm* existiert. Das Sysop-Passwort wird in der Datei *passwd.bcm* abgelegt. Falls das AX25-Passwort die Zeichenkette "DUMMY" (in Grossbuchstaben) enthält, ist ein Login via HTTP/POP3 trotzdem möglich. Dort wird dann als Passwort der Vorname abgefragt, der zuvor mit ALTER NAME über das AX25-Login gesetzt wurde. SMTP ist im Allgemeinen nur möglich, wenn zuvor POP3 ausgeführt wurde oder ALTER UNSECURESMTP 1 gesetzt ist.

12. PW & Hold/Reject functions

Seit dem Jahr 2000 ist die OpenBCM mit einem erweitertem PW & HOLD/REJECT-System ausgestattet, doch leider wird dies bislang so gut wie gar nicht angewendet.

PW & HOLD/REJECT ist ein zweiteiliges System, mit dem sich die Sicherheit der Benutzer gegen Rufzeichenmissbrauch bedeutend erhöhen lässt.

Dies geschieht durch zweierlei Maßnahmen:
- Schutz für Benutzer, die noch kein Passwort benutzen
- Hold-Mechanismus für Nachrichten aus ungeschützter Quelle

a) Passwortverfahren
Das geänderte Passwortverfahren wird mit dem Parameter "USERPW 2" aktiviert.

Damit gelten automatisch eine Reihe Schutzmassnahmen für Benutzer ohne Passwort, was den Zugriff auf Nachrichten betrifft:

- kein unauthorisiertes Löschen
- kein Schreiben von (gefälschten) Nachrichten mit Forward
- kein Lesen von fremden Usermails
- kein Verstellen von wichtigen Parametern (wie MyBBS)

In der Datei *runutil.bcm* dient der Parameter "-P" dazu, dass Programme nur von Benutzern mit User-PW aufgerufen werden können.

In der Datei *msg\pwonly.<sprache>* kann man einen Hinweistext für Benutzer ohne Passwort hinterlegen, der beim Login gesendet wird. Hier sollte man insbesondere darauf hinweisen, wie man ein Passwort bekommt.

Im Gegensatz zum bisherigen Verhalten, wo unter Rufzeichenmissbrauch jeder jedem ein Passwort setzen (und somit aussperren) kann, wird das Passwort nun vom Sysop eingerichtet. Mit den Befehlen PWGEN und SETPW lässt sich eine automatische Passwort-Vergabe durch den Sysop einrichten:
Dazu wird mittels PWGEN eine Datei *userpw.bcm* mit 81 Passwörtern vorbereitet, die dann mit "setpw <user> <nr>" Usern zugeordnet werden können. Diese Datei *userpw.bcm* sollte man auch zu Hause haben, um das zugeteilte Passwort dem User mitteilen zu können (natürlich nicht über Funk auslesen!).

Mit dem Parameter "USERPW 3" wird das User-Passwort nicht mehr beim Login, sondern erst später abgefragt - analog zur DPBox. So wird das User-Passwort nur noch zum Schreiben und Löschen von Mails sowie zum Verstellen von Einstellungen benötigt. Vorteil: ein vergessenes Passwort sperrt einen nicht gleich aus.

b) HOLD-Mechanismus
Der HOLD/REJECT-Mechanismus wird über die Datei *reject.bcm* gesteuert. Zum bequemen Verwalten der Einstellungen gibt es den REJECTEDITOR. In *reject.bcm* kann man parametrieren, wie mit Nachrichten verfahren wird, die ohne Passwort gesendet werden oder über ungeschützte S&F-Wege hereinkommen. Mindestens ein

Eintrag "P .B" (kein Bulletin-Send ohne AX25-PW) sollte vorhanden sein. Alle Optionen, die in *reject.bcm* angewendet werden können, sind im Kapitel "5.10. Reject and Hold (reject.bcm)" beschrieben.

Wie lange angehaltene Nachrichten nicht geforwarded werden, wird über den Parameter HOLDTIME in Stunden eingestellt (typisch: 48 Stunden = 2 Tage).

Mit DIR HOLD lassen sich angehaltene Nachrichten aufzeigen. DIR HOLD funktioniert genauso wie DIR NEWS oder CHECK abhängig vom LastCheck-Timer.

Mit "HOLD -u" bzw. "FORWARD -h" kann der Sysop auf Hold gesetzte Mails wieder in den Forward geben.

13. M(ail)filter function

Es ist möglich mit Hilfe eines externen (Mailfilter-)Programms eingegebene Mails in der Mailbox auf bestimmte Dinge hin zu untersuchen, und dann resultierend vom Ergebnis eine Aktion mit der Mail ausführen.

Das externe Programm kann vom Sysop mit dem Befehl M_FILTER <Programm> eingestellt werden.

Das <Programm> kann dann die Nachricht untersuchen, z.B. auf unerwünschte Schlüsselwörter, oder ähnliches. Das <Programm> muss einen Rückgabewert übermitteln, von dem abhängig dann folgendes geschieht:
0: es passiert nichts
1: die Nachricht wird gelöscht
2: die Nachricht wird nicht weitergeleitet (Hold)

Unter Linux wird der Rückgabewert nicht ausgewertet, hier kann M_FILTER also nur dazu verwendet werden, ein externes Programm aufzurufen.

Der M_FILTER Aufruf wird in der Datei *trace\m_filter.bcm* protokolliert.

Anmerkung: Ist das <Programm> nicht vorhanden, oder ist der Pfad falsch eingestellt, kommt es beim Senden von Mails zu einer Ausgabe im Syslog. Ferner könnte unter Linux auch noch die Möglichkeit bestehen, dass die Dateizugriffsberechtigungen des M_Filter-Programmes falsch gesetzt sind (es muß mindestens 'rx' gesetzt sein). Um das Mailfilterprogramm zu deaktivieren muß nur die Option M_FILTER OFF in der *init.bcm* gesetzt werden.

Hinweis: Der Funkrufserver unter Linux von Holger Flemming, DH4DAI und Jens Schoon, DH6BB (siehe im Internet "http://sourceforge.net/projects/ham-pager") verwendet ebenfalls die M_FILTER Schnittstelle. Der Pager gibt jedoch immer 0 (es passiert nichts) an die Mailbox zurück.

14. Backup of mailbox data

14.1. Backup using DOS

In älteren Ausgaben der Dokumentation wurde empfohlen ein Backup mittels eines Batches zu machen. Genau dieses Backup führte aber oft zu Problemen (User werden disconnected weil Flexnet zu lange nicht gepollt wurde, Box gerät in den Watchdog-Reset weil der DOS-Aufruf zu lange dauerte etc.).

Deshalb ist es besser, gelegentlich ein Backup der gesamten Platte zu machen als täglich Dateien umzukopieren. Hier ist z.B. die kommerzielle Software "Norton Ghost" oder "Powerquest Drive Image" zu empfehlen.

Generell gilt: Die Dateien u*sers4.bcm* und *hadr4.bcm* werden zwar von der Mailbox erzeugt, können aber nicht aus anderen Daten restauriert werden, falls sie verlorengehen. Sie sollten deshalb regelmäßig gesichert werden.

14.2. Backup using Linux

Unter Linux kann ein "save"-Aufruf für *save.imp* im *crontab.bcm* angeführt werden. Ein *save.imp* kann etwa folgendermaßen aussehen:

```
oshell mkdir backup
oshell mv -f backup\bu002.tgz backup\bu003.tgz
oshell mv -f backup\bu001.tgz backup\bu002.tgz
oshell tar zcvf backup\bu001.tgz *.bcm
```

Während der Abarbeitung des Backups läuft die Mailbox im Gegensatz zu DOS ohne Unterbrechung weiter.

Auch hier ist ein kommerzielles Partitions-Backuptool wie z.B. "Powerquest Drive Image 2002" zu empfehlen, das auch Linux-Partitionen backup-pen kann. Dann erspart man sich im Festplatten-Fehlerfall auch die Neuinstallation und vor allem Neukonfiguration des kompletten Linuxsystems incl. Mailbox.

14.3. Backup using Windows

Unter Windows kann ebenso wie bei Linux ein "save"-Aufruf im *crontab.bcm* angeführt werden. Die Datei *save.imp* kann etwa folgendermaßen aussehen:

```
oshell mkdir backup
oshell move backup\bu002.zip backup\bu003.zip
oshell move backup\bu001.zip backup\bu002.zip
oshell pkzip -9 backup\bu001.zip *.bcm
```

Anstelle von *pkzip.exe* kann natürlich auch ein anderes Packprogramm verwendet werden. Da kein Packprogramm Bestandteil des Betriebssystems Windows ist, muss dies auf alle Fälle nachträglich installiert werden. Zum fehlerfreien Aufruf bietet es sich an, den Packer in die PATH-Variable aufzunehmen.

Während der Abarbeitung des Backups läuft die Mailbox im Gegensatz zu DOS ohne Unterbrechung weiter.

Auch unter Windows ist ein kommerzielles Partitions-Backuptool wie z.B. "Powerquest Drive Image 2002" zu empfehlen. Dann erspart man sich im Festplatten-Fehlerfall auch die Neuinstallation und vor allem Neukonfiguration des kompletten Betriebssystems incl. Mailbox.

15. Timed processes

Gesteuert wird die gesamte Zeitverarbeitung durch die Datei *crontab.bcm*. Diese Datei ist im Format sehr ähnlich der crontab-Datei unter UNIX/Linux.

Die folgende Beispiel-Datei zeigt den Aufbau für zeitgesteuerten Jobs:

```
# Timing-File for OpenBCM-Mailbox - CRONTAB.BCM
#
# ranges:
# minute: 0-59    hour: 0-23     day: 1-31     month: 1-12
# weekday: 0-6    (0:sun 1:mon 2:tue 3:wed 4:thu 5:fri 6:sat)
#
#minute   hour  day month weekday   command
*          *     *   *     *       minute
```

```
*/5          *    *    *     *       fiveminute
0,30         *     *   *     *       halfhour
1,31         *     *   *     *       beacon
3,33         *     *   *     *       forward
5            2     *   *    1,4      postfwd
5            3     *   *     *       purge
35           3     *   *     *       save
5            4     *   *     2       reorg
0,15,30,45   *     *   *     *       quarter
2            0     *   *     *       hour0
2            1     *   *     *       hour1
[...]
2           22     *   *     *       hour22
2           23     *   *     *       hour23
```

Kommentarzeilen beginnen mit einem Zeichen Strichpunkt (;) oder einer Raute (#) in der ersten Spalte, jedes Zeichen ungleich 0-9 oder * ist ebenfalls zur Kennzeichnung eines Kommentars zulässig.

In den linken 5 Spalten werden Zeiten angegeben. Hier kann jeweils entweder eine Zeit stehen, oder mehrere durch "," getrennt, oder Zeitbereiche in der Form "5-12" oder jede beliebige Kombination die nicht länger als 30 Zeichen ist, also z.B. "1,2,4-7,5,6-12,16". Ein "*" steht für "jede Zeit ist gültig". Mit z.B. "*/5" ist es möglich z.B. alle 5 Minuten Aufgaben ausführen zu lassen. Als Zeitpunkt werden Minuten, Stunden, Tag im Monat, Monat und Tag der Woche angegeben. Bei letzterem bedeutet 0=Sonntag und zählt bis 6=Samstag. Alle Zeitangaben müssen erfüllt sein, damit das angegebene Programm ausgeführt wird ("UND"-Verknüpfung). Sollten mehrere Zeitmodelle "ODER"-verknüpft werden, so kann dasselbe Programm in mehreren Zeilen angeführt werden.

Das ausgeführte Kommando ist entweder eine Batch-Datei (wird auf Betriebssystem-Ebene ausgeführt und muss im Dateisystem die Erweiterung *.bat* haben), oder eine Import-Datei (wird auf */bcm*-Ebene ausgeführt und muss im Dateisystem die Erweiterung *.imp* haben). Außerdem werden einige reservierte, interne Kommandos ausgeführt. Diese sind:

purge
: Startet das Ausputzen des kompletten Dateisystems (wurde bis zur Version 1.34 zu "purgehour" gestartet).

forward
: Stößt das Forwarding zu benachbarten Boxen an (wurde bis zur Version 1.34 zur vollen und halben Stunde gestartet).

beacon
: Stößt das Erzeugen einer Mail-Bake an (wurde bis zur Version 1.34 in "mailbeacon"-Intervallen gestartet).

postfwd
: Leitet liegen gebliebene Usermails weiter.

reorg [opt]
: Startet die Dateisystem-Reorganisation. Wenn [opt] weggelassen wird, wird ein kompletter Reorg gemacht. Ansonsten kann ein Teil-Reorg angegeben werden.

Sind mehrere Bedingungen in *crontab.bcm* gleichzeitig erfüllt, so werden sie in der Reihenfolge der *crontab.bcm* abgearbeitet. Wenn unter DOS ein Batch-Programm (also ein externer DOS-Aufruf) mehr als eine Minute dauert, kann es passieren dass andere Jobs, die unter derselben Zeit, aber weiter hinten stehen, nicht mehr aufgerufen werden, weil inzwischen der Aufrufzeitpunkt überschritten wurde. Unter Linux tritt dieses Problem nicht auf, da die Ausführung der Mailbox durch einen Shell-Aufruf nicht unterbrochen wird.

Interne Kommandos (PURGE, FORWARD, etc.) sowie IMPORT-Dateien werden nebenläufig ausgeführt, d.h. blockieren sich nicht gegenseitig, sondern starten wirklich zur gleichen Zeit. Wenn solche Überschneidungen unerwünscht sind, so ist dies durch entsprechendes Timing oder SLEEP-Kommandos in den entsprechenden Import-Dateien zu verhindern.

Die Namen der BATches und IMPorts brauchen nicht "hour17" zu lauten, es ist sinnvoller der Anwendung entsprechende Namen zu verwenden, etwa *mkmstat.imp* für "make-monthly-statistics".

16. Automatic server

16.1. Filesurf

The OpenBCM mailbox contains a file server (or shortly named filesurf). This server can be used to easy exchange files. Comparing with bulletin mails it has following advantages:

- The user can define for each download which output protocol should be used (AutoBIN, YAPP, Didadit etc.)
- The files may better sorted, because directories can be used
- The mailbox management is not stressed, because the filesurf files must not be managed by the mailbox itself (no lifetime, no forward, no BID etc.)

Um die Dateien zur Verfügung zu stellen, müssen sich die Dateien unterhalb eines oder mehrerer Verzeichnisse auf der Festplatte befinden. Diese Verzeichnisse werden dann durch den Parameter FSPATH in der Datei *init.bcm* für die Benutzer freigegeben. Dabei werden die verschiedenen Pfade jeweils durch ein Leerzeichen getrennt. Möchte man also z.B. die Verzeichnisse */filesurf* und */cdrom* freigeben, so gibt man in der Mailbox einfach

```
FSPATH /filesurf /cdrom
```

ein. Dabei ist zu beachten, daß grundsätzlich alle Dateien und Unterverzeichnisse unterhalb dieser Verzeichnisse freigegeben werden. Unter Linux ist allerdings zusätzlich darauf zu achten, daß der User "bcm" auf die gewünschten Dateien Zugriff hat.

Unter DOS und Windows dürfen auch Laufwerksbuchstaben angegeben werden, z. B.:

```
FSPATH E:\ D:\FILESURF
```

Der Benutzer kann dann wie an der DOS-Kommandozeile üblich mit dem Laufwerksbuchstaben gefolgt von einem Doppelpunkt (also z.B. *E:*) auf das gewünschte Laufwerk wechseln, dabei wird dann in das erste Verzeichnis gewechselt, das auf dem gewählten Laufwerk freigegeben ist.

Will man den Benutzern den Schreibzugriff auf ein Verzeichnis gewähren, so ist dem Verzeichnis ein "+" voranzustellen. Um gegenüber obigem Beispiel also z.B. noch das Verzeichnis */user* mit Schreibzugriff freizugeben, ist folgendes einzugeben:

```
FSPATH /filesurf /cdrom +/user
```

To start the filesurf, simply type in the command FS. You can exist the filesurf mode with command QUIT. The filesurf commands are simple and the look and feel is similar a DOS or Linux shell. You can find a list of all commands in chapter "25.4. Overview of all filesurf commands".

Note: The filesurf is only available, if the filesurf option has been compiled into in the mailbox executable!

When you are running OpenBCM with Linux or Windows, the filesurf can also be reached via the FTP server, default located at port 8021 (see also chapter "19.6. FTP access using Linux and Windows" for this).

16.2. Mailserver

The OpenBCM mailbox contains a mailing list server (or shortly named mailserver).

The mailserver can be used to send usermails to more than one user at one time.

If a mail should be send to a mailserver group, the mail must be addressed to the mailbox callsign and the title of the mail must have following syntax:

```
mailto <mailserver group> <title>
```

For configuration of the mailserver following commands can be used:

```
MAILServer +G      : add a new mailserver group
MAILServer -G      : delete a mailserver group
MAILServer Newgroup: add a new mailserver group
MAILServer Delgroup: delete a mailserver group
MAILServer DEScrip : define a description of a mailserver group
MAILServer Options : set options of a mailserver group
MAILServer +U      : add a callsign to a group
MAILServer -U      : delete a callsign from a group
MAILServer ADDUser : add a callsign to a group
MAILServer DELUser : delete a callsign from a group
MAILServer +M      : add a maintainer to a group
MAILServer -M      : delete a maintainer from a group
MAILServer ADDMain : add a maintainer to a group
MAILServer DELMain : delete a maintainer from a group
MAILServer Info    : show info of a group
MAILServer List    : list all available mailserver groups
MAILServer SUbscr  : subscribe own callsign to a group
MAILServer Unsubs  : unsubscribe own callsign from a group
MAILServer Reset   : reset the mail counter to a value of 0
MAILServer Setnum  : set the mail counter to a special number
```

With sysop command MAILLISTSERV the mailserver can be activated/deactivated:

```
maillistserv 0   deactivated
maillistserv 1   activated
maillistserv 2   activated, only the sysop can add new mailserver groups
```

Note: The mailserver is only available, if the mailserv option has been compiled into in the mailbox executable!

16.3. Configuration of external automatic server

Verschiedene Funktionen (z.B. das Einlesen der aktuellen Wetterwerte von einer Wetterstation) können regelmäßig von der Box durchgeführt werden. Dazu stehen periodisch aufgerufene Skript-Dateien zur Verfügung. Der Name und der Zeitpunkt des Aufrufs des Batch- bzw. Import-Dateien wird durch die Datei *crontab.bcm* gesteuert.

Es werden jeweils 2 Dateien mit der Endung *.bat* und *.imp* ausgeführt. Die *.bat*-Datei wird dabei dem Kommandointerpreter des jeweiligen Betriebssystems übergeben. Die *.imp*-Datei wird innerhalb der Box ausgeführt, d.h. in dieser Datei können Boxbefehle wie "send" oder "ps" enthalten sein.

Dabei wird immer zuerst die BATch-Datei ausgeführt und anschließend die dazugehörige IMPort-Datei. Existiert eine in *crontab.bcm* angeführte *.bat*- oder *.imp*-Datei nicht, so hat das keine Folgen. Es kann zu einem Eintrag in *crontab.bcm* auch nur eine *.bat*- oder *.imp*-Datei existieren, oder keines der beiden. Während der Ausführung des BATCH-Jobs "steht" die DOS-Box, es ist darauf zu achten, dass es durch oft aufgerufene DOS-Batch-Jobs nicht zu allzu großen Verzögerungen im Box-Betrieb kommt, und dass ein einzelner DOS-BATCH-Job nicht länger als 30 min dauert. Unter Linux und Windows besteht diese Zeitbegrenzung nicht, allerdings ist hier darauf zu achten, dass mehrere Batches eventuell gleichzeitig aufgerufen werden.

Bei der Abarbeitung der Import-Dateien wird ein Login mit dem Boxnamen mit Sysopstatus durchgeführt. Es können also alle Befehle ausgeführt werden, die ein Sysop ausführen darf. Zum Speichern einer Nachricht unter einem bestimmten Rufzeichen ist also z.B. folgendes zu schreiben:

```
call dl2ja
e fsg_wx 1-
s fsg_wx @ db0fsg neues Wetter
..blabla..
nnnn
impdel halfhour.imp
```

Denkbar ist, dass von *halfhour.bat* (oder welchem **.bat* auch immer) eine Import-Datei erzeugt wird, die dann anschließend eingelesen wird. Innerhalb eines Import-Vorgangs ist sowohl ein Export möglich, als auch ein verschachtelter Import. Damit ist es auch möglich, dass eine Import-Datei durch OSHELL-Befehle im Import-Prozess erzeugt wird, und anschließend im selben Import-Prozess noch importiert wird.

Unbedingt zu beachten ist, dass ein Import-Prozess, der innerhalb eines anderen Import-Prozesses erzeugt wird, gleichzeitig zum Mutterprozess abläuft. Ist der Mutterprozess also von einem Ergebnis des Tochterprozesses abhängig, so muss diesem durch ein SLEEP-Kommando Folge getragen werden. Außerdem ist das Prozess-System der OpenBCM-Mailbox flach, d.h. wenn der Mutterprozess beendet wird, läuft der Tochterprozess evtl. noch weiter.

Zum Testen von Batch- und Import-Dateien dient der Befehl BATCH. Mit diesem kann der Vorgang angestoßen werden. Die Endung der Dateien wird nicht angegeben:

```
batch save     führt save.bat und save.imp aus
batch hour17   führt hour17.bat und hour17.imp aus
```

Die Ausgaben der Batch-Dateien werden unter DOS in gleichnamige Dateien mit der Erweiterung *.OUT* bzw. *.ERR* umgeleitet. Die Ausgabe der Batch-Datei *hour.bat* erfolgt also z.B. in die Dateien *hour.out* und *hour.err*.

17. Mail beacon

In der Mail-Bake werden alle Rufzeichen gesendet, für deren Inhaber eine Nachricht vorhanden ist, die von ihm noch nicht gelesen wurde. Das Rufzeichen wird dabei nicht in die Liste aufgenommen, wenn der Benutzer nach dem Eintreffen der neuen Nachricht bereits eingeloggt war. Außerdem werden nur Mails berücksichtigt, die nicht älter sind als mit dem Parameter OLDESTBEACON in der Datei *init.bcm* angegeben (die Angabe erfolgt in Tagen). Die Aussendung eines Rufzeichens erfolgt unabhängig davon, ob es sich um die Heimatmailbox des Benutzers handelt.

Sind ungelesene Nachrichten für mehr User vorhanden, als in einer Bake Rufzeichen untergebracht werden können, so werden mehrere Mail-Baken generiert.

Die Zeitpunkte der Ausstrahlung der Mail-Bake werden durch die Datei *crontab.bcm* eingestellt. Soll die Mail-Bake alle 20 Minuten ausgesendet werden, so ist folgender Eintrag sinnvoll:

```
[...]
   0,20,40    *    *    *    *    beacon
[...]
```

Zur Konfiguration der Dateien *beacon.bcm* und *beachead.bcm* siehe auch das Kapitel "5.9. Beacon configuration (beacon.bcm *and* beachead.bcm)".

Über den Sysopbefehl MAILBEACON kann man die Mailbaken-Ausstrahlung auch dann erzwingen, wenn keine neuen Usermails vorliegen. Dies sollte im Regelfall aber keinen Sinn machen!

18. External programs

Von der Mailbox aus können externe Programme aufgerufen werden. Bestimmte Programme können für Benutzer freigegeben werden, andere sind nur mit Sysopstatus auszuführen.

18.1. Sysop execution of external programs

Der Aufruf von einem beliebigen externen Programm erfolgt für den Sysop durch die Eingabe von

OSHELL <Programm-Name> <Parameter>

am Mailbox-Prompt.

18.2. User execution of external programs

Um ein Programm für Benutzer zugänglich zu machen, muss das Programm in einer Datei namens *runutil.bcm* eingetragen sein. Siehe hierzu auch das Kapitel "5.13. Runutils (runutil.bcm)".

18.3. Requirements for an external program

Das aufgerufene Programm muss folgende Eigenschaften haben:

Unter DOS:

- Schnelle Ausführung! Währenddessen steht die Box für alle User! Ein Run-Utility sollte auf keinen Fall länger als etwa eine Sekunde zur Ausführung benötigen.
- Ausgaben nur über DOS/BIOS, nicht direkt in den Bildschirmspeicher.
- Keine Eingaben! Wenn das Programm auf eine Eingabe wartet, steht alles still.
- Geringer Speicherbedarf. Es steht i.A. nur wenig freier Speicher zur Verfügung (typisch ca. 100 kB, siehe Anzeige im V-Befehl "Heap")
- Es kann sich um eine COM-, EXE- oder BAT-Datei handeln.
- Der Aufruf muss "wasserdicht" sein, d.h. der User darf mit einer x-beliebigen Kommandozeile keinen Schaden anrichten können.

Unter Linux:

- Die Box läuft während der Ausführung des Run-Utilities weiter.
- Eingaben an Run-Utilities werden akzeptiert.
- Gleichzeitige Ausführung eines Run-Utilities durch mehrere User ist möglich.
- Speicherbedarf unkritisch.
- Der Aufruf muss "wasserdicht" sein, d.h. der User darf mit einer x-beliebigen Kommandozeile keinen Schaden anrichten können.

Unter Windows:

- Die Box läuft während der Ausführung des Run-Utilities weiter.
- Ausgaben nur über DOS/BIOS, nicht direkt in den Bildschirmspeicher.
- Es kann sich um eine COM-, EXE- oder BAT-Datei handeln.
- Gleichzeitige Ausführung eines Run-Utilities durch mehrere User ist möglich.
- Der Aufruf muss "wasserdicht" sein, d.h. der User darf mit einer x-beliebigen Kommandozeile keinen Schaden anrichten können.

18.4. Parameters for the run utility

Unter DOS und Windows werden die angegebenen Parameter bis auf die Zeichen "<", ">" und "|" transparent an das Run-Utility übergeben. Ist eines der genannten Zeichen enthalten, wird der Parameter-String an dieser Stelle abgeschnitten.

Unter Linux wird allen Zeichen ein Backslash ("\") vorangestellt, so dass von der Shell kein Zeichen anderweitig interpretiert wird. Auf diese Weise können auch die Zeichen "<", ">" und "|" übergeben werden. Letzteres geschieht aus Sicherheitsgründen aber nur, wenn dies in *runutil.bcm* mit dem Parameter "-T" eingestellt wurde.

Auf jeden Fall ist es einfacher, wenn ein Run-Utility die in *rundat.bcm* unter "cmdline" (siehe unten) angegebene Kommandozeile auswertet. Hier sind sogar mehrere Leerzeichen zwischen zwei Parametern sichtbar. Allerdings muss natürlich auch das Run-Utility aufpassen, wenn es mit den Parametern weitere Shell-Aufrufe macht.

18.5. Syntax of file *rundat.bcm*

Die Datei *rundat.bcm* wird von der OpenBCM-Mailbox vor jedem Run-Utility-Aufruf erzeugt, außer es wurde die Option "-Q" in *runutil.bcm* angegeben.

Achtung: Unter Linux/Windows ist diese Datei nur unmittelbar nach dem Aufruf gültig. Zu einem beliebigen späteren Zeitpunkt kann diese Datei bereits die Daten für einen anderen Run-Utility-Aufruf enthalten.

In der Datei *rundat.bcm* sind einige Informationen über den Benutzer gespeichert, der das Run-Utility aufgerufen hat. Das externe Programm kann diese Informationen entsprechend auswerten.

Die erste Zeile enthält einige Angaben zum Benutzer in folgender Reihenfolge:

- Call ohne SSID
- MYBBS
- Name
- Mit "ALTER SPEECH" eingestellte Sprache
- 1, wenn der Login mit Passwort erfolgt ist, sonst 0
- 1, wenn der Benutzer Sysop-Status hat, sonst 0
- Status (siehe ALTER STATUS)
- Helplevel (siehe ALTER HELPLEVEL)
- Einstiegsdigi (nur richtig, wenn der Connect ausschließlich über Flexnet-Digis erfolgt ist)

Diese Informationen sind nur durch Leerzeichen getrennt und müssen anhand der Reihenfolge ausgewertet werden. Zu beachten ist, dass die Informationen in der ersten Zeile nur aus Kompatibilitäts-Gründen vorhanden sind. Sie sollte nicht mehr verwendet werden, weil diese erste Zeile in späteren Versionen der OpenBCM-Mailbox evtl. nicht mehr erzeugt wird.

Ab der zweiten Zeile folgen Informationen in der Form "Schlüsselwort Wert". Die Schlüsselwörter haben folgende Bedeutung:

call
: Call ohne SSID

mybbs
: MyBBS (ALTER FORWARD)

name
: Name (ALTER NAME)

language
: Sprache (ALTER SPEECH)

password
1, wenn der Login mit Passwort erfolgt ist, sonst 0

sysop
1, wenn der Benutzer Sysop-Status hat, sonst 0

status
User-Status (ALTER STATUS)

helplevel
Help-Level (ALTER HELPLEVEL)

digi
Einstiegsdigi (nur richtig, wenn der Connect ausschließlich über Flexnet-Digis erfolgt ist)

txt_output
Datei, die nach Beendigung des Run-Utilities als Text ausgegeben wird

bin_output
Datei, die nach Beendigung des Run-Utilities binär - allerdings ohne AutoBIN-Header - ausgegeben wird

abinoutput
Datei, die nach Beendigung des Run-Utilities im AutoBIN-Format ausgegeben wird

yappoutput
Datei, die nach Beendigung des Run-Utilities im YAPP-Protokoll ausgegeben wird

imp_output
Datei, deren Inhalt nach Beendigung des Run-Utilities im Hintergrund ausgeführt wird; die Abarbeitung der Befehle geschieht mit Sysop-Status und als Prozess-Rufzeichen wird das der Mailbox verwendet. Sind die letzteren beiden Eigenschaften nicht gewünscht, so kann

- das Rufzeichen in der IMPORT-Datei mit "call <rufzeichen>" geändert werden und
- der Sysopstatus mit "PW OFF" verlassen werden.

Soll diese Datei nach der Ausführung gelöscht werden, so ist an beliebiger Stelle der Befehl "IMPDEL" ohne Parameter einzufügen.

user_imp
Datei, deren Inhalt nach Beendigung des Run-Utilities so abgearbeitet wird, als wären es Benutzereingaben gewesen

cmd_line
Befehlszeile des Benutzers, mit der der Aufruf des Run-Utilities erfolgte

login_time
Loginzeit im Unix-Format (Sekunden seit 1.1.1970)

board
Aktuelles Board

fspath
Aktueller Pfad im Filesurf

Die Auswertung der Informationen sollte unbedingt anhand der Schlüsselwörter erfolgen, und nicht anhand der Reihenfolge. Die Dateinamen bei den Schlüsselwörtern txt_output, bin_output, abinoutput, yappoutput, imp_output und user_imp sind eindeutige, temporäre Namen, die von der Box vorgegeben werden. Das Run-Utility sollte hier auch unbedingt längere Pfadnamen akzeptieren, damit künftig problemlos Änderungen auf Boxseite möglich sind.

Achtung:
Auch unter DOS und Windows werden als Pfadtrenner Slashes ("/"), und keine Backslahes ("\") verwendet! Mit Ausnahme von "imp_output" werden alle Dateien nach der Ausgabe bzw. nach der Bearbeitung automatisch gelöscht.

Beispiel einer *rundat.bcm*:

```
DGT274 DBO274.#NRW.DEU.EU Markus DL 0 0 0 0 DNX274
call      DGT274
mybbs     DBO274.#NRW.DEU.EU
name      Markus
```

```
language   DL
password   0
sysop      0
status     0
helplevel  0
digi       DNX274
txt_output temp/cctfci0
bin_output temp/cctfci1
imp_output temp/cctfci4
user_imp   temp/cctfci5
cmdline    7m
login_time 1041175316
board      DGT274
abinoutput temp/cctfci2
yappoutput temp/cctfci3
fspath     /filesurf/openbcm/beta/
```

Die Liste kann ggf. nach unten erweitert werden.

Nach Beendigung des Run-Utilities erfolgt die Bearbeitung der vom Run-Utility erzeugten Dateien und Ausgaben in folgender Reihenfolge (falls eine Datei nicht erzeugt wurde, geschieht einfach gar nichts):

- Ausführung von imp_output (erfolgt im Hintergrund, eigene Task)
- Ausgabe der STDOUT-Ausgaben des Run-Utilities
- Ausgabe der STDERR-Ausgaben des Run-Utilities
- Ausgabe von txt_output als normaler ASCII-Text
- Binäre Ausgabe von bin_output (transparent, d.h. ohne jeden Vor- und Nachspann)
- Ausgabe von abinoutput im AutoBIN-Format
- Ausgabe von yappoutput im YAPP-Protokoll
- Ausführung der Befehle in der Datei user_imp

Wie an den vielen Dateizugriffen zu sehen ist, ist für einen befriedigend schnellen Betrieb von Utilities ein ordentlicher Rechner notwendig.

19. Non-AX.25 access of OpenBCM

19.1. Using a serial device in the DOS version

Diese Option ist normalerweise unter DOS wegen Speichermangels nicht mehr verfügbar.

Die Box wird bei aktivierter Option über einen seriellen Port angesprochen. Dabei wird ein primitives Terminal (z.B. VT100 oder XTALK) benötigt und man erhält eine Benutzeroberfläche wie über Funk. Zum Login ist allerdings eine Anmeldung mit Rufzeichen und Passwort erforderlich. Das Passwort kann nur von einem Sysop mit

```
SETUSER <Rufzeichen> TTYPW <Passwort> (max. 7 Zeichen)
```

eingegeben werden und ist für jeden Benutzer einzeln zu vergeben. Ohne vorher eingetragenes Passwort kann man sich nicht über die serielle Schnittstelle einloggen. Dies ermöglicht, dass auch Unbefugte Zutritt zum Terminal haben können, ohne die Datensicherheit der Box zu gefährden. Beim Passwort wird Groß- und Kleinschreibung unterschieden. Durch die Identifizierung mit dem TTYPW ist man nicht automatisch Sysop der Mailbox, dazu ist eine zusätzliche Sysop-Passwort-Identifizierung notwendig.

Zur Konfiguration ist mit TTYMODE folgender Parameter einzustellen(DOS):

```
TTYMODE 1:9600,n,8,1,e
                    ^e=echo ein, l=echo lokal
                  ^Anzahl der Stopbits
                ^Anzahl der Datenbits
```

```
                ^Parity n=no e=even o=odd
            ^baudrate 1200-38400
          ^com-Schnittstelle COM1-COM4
```

Standard-COM-Adresse/-Interrupts:

- COM1: IRQ 4, Base-Addr 0x3f8
- COM2: IRQ 3, Base-Addr 0x2f8
- COM3: z.B. IRQ 5 (nicht standardisiert), Base-Addr 0x3e8
- COM4: z.B. IRQ 7 (nicht standardisiert), Base-Addr 0x2e8

Ist ein 16550 (Baustein mit Puffer) eingebaut, so wird dieser erkannt und benutzt. Als Handshake wird stets XON/XOFF verwendet. Es sind deshalb neben den Datenleitungen (GND, TXD, RXD) auch die Handshakeleitungen zu verbinden.

Bei der Anwendung dieser Schnittstelle ist der Phantasie freier Lauf gelassen. Denkbar wäre z.B. eine Anbindung an ein Modem. Das Abschalten des TTY-Ports erfolgt mit TTYMODE OFF (Default).

Wichtig: Die Installation des TTY-Ports wird erst beim nächsten Start der DOS-OpenBCM-Mailbox durchgeführt.

19.2. TELNET access using Linux and Windows

Die Mailbox kann unter Linux/Windows über Telnet angesprochen werden.

Diese Funktion macht aufgrund der hohen Laufzeiten über Funk wenig Sinn. Jedoch innerhalb eines drahtgebundenen Netzes stellt dies die beste Möglichkeit dar, die OpenBCM-Mailbox zu erreichen.

Die OpenBCM-Mailbox erwartet normalerweise auf dem TCP Port 4719 einen Telnet Verbindungsaufbau. Dieser Port kann in *init.bcm* (Parameter TELNET_PORT) geändert werden. Die Erreichbarkeit über Telnet kann auch durch Setzen dieses Wertes auf 0 ganz abgeschaltet werden.

Folgen wir zur Tat:

```
bash$ telnet server 4719
Trying 172.16.1.1...
Connected to server.local.net.
Escape character is '^]'.

OpenBCM-Mailbox v1.05 (Linux)
DH8YMB login: dh8ymb
password: xxxxxxx

OpenBCM-Mailbox v1.05 (Linux) - Silly Valley - 01:10z
Hallo Markus, Helplevel=2, Zeilen=0, letzter Login 24.07.02 12:12z

Inhaltsverzeichnis für DH8YMB @ DH8YMB:
372) DH8YMB 24.11.02 11:03 13 #999 @ DH8YMB OpenBCM Demo
(12:13 DH8YMB)-->
```

Ob ein Passwort und welches Passwort eingegeben werden muss, hängt dabei von folgenden Aspekten ab:

- *rhosts.bcm* - In dieser Datei werden alle 'trusted hosts' eingetragen, damit sind die Namen oder IP-Adressen von Rechnern gemeint, denen vertraut werden kann und die dann gleich automatisch Sysopstatus besitzen, siehe hierzu auch Kapitel "5.7. Remotehost (rhosts.bcm)".
- Bei allen anderen Logins kann das Passwort nur durch den Sysop gesetzt werden. Es gilt dabei der Parameter (A TTY). Nach der korrekten Eingabe des Passwortes ist der Benutzer eingeloggt und besitzt keinen Sysopstatus.

Das Auslesen von Binärmails in einer Telnet-Session ist im Allgemeinen nicht besonders sinnvoll.

19.3. HTTP access using Linux and Windows

Im Gegensatz zu den meisten anderen Lösungen (CGI-Skript), besitzt die OpenBCM-Mailbox ein eingebautes HTTP Interface.

Dadurch lässt sich die Box voll interaktiv bedienen. Das bedeutet, Befehle werden direkt ausgeführt und Benutzer können sich unter ihrem eigenen Rufzeichen einloggen.

Der HTTP Server ist normalerweise auf dem TCP Port 8080 erreichbar. Andere Werte lassen sich über *init.bcm* (Parameter HTTP_PORT) setzen. Die Erreichbarkeit über HTTP kann auch durch Setzen dieses Wertes auf 0 ganz abgeschaltet werden.

Bei der ersten HTTP Verbindung zu einer OpenBCM-Mailbox kommt ein kleines Fenster auf den Bildschirm. Dort wird man aufgefordert einen Namen und ein Passwort einzugeben. Auch hier gelten ein paar Konventionen bezüglich der Eingaben: Als Name wird grundsätzlich das eigene Rufzeichen eingegeben. Beim Passwort trifft die OpenBCM-Mailbox eine kleine, aber wichtige Unterscheidung:

- Wird eine Verbindung vom localhost [127.0.0.1] oder dem Packet Radio Netz (44.x.x.x.) aufgebaut, dann gilt als Passwort der in der OpenBCM-Mailbox eingegebene Name des Benutzers, sofern in *init.bcm* nicht HTTPTTYPW auf 1 gesetzt wurde. Durch diese Konvention muss den Benutzern durch den Sysop kein Passwort eingetragen werden. Hat ein Benutzer ein AX.25-Password gesetzt, ist kein Login möglich, es sei denn, der Benutzer hat in dem AX25-Passwort-String die Buchstabenkette "DUMMY" mit eingefügt, dann wird hier ebenfalls der Name des Benutzers abgefragt. Ist HTTPTTYPW auf 1 gesetzt, so wird generell nur das Passwort, das unter ALTER TTYPW gesetzt wurde, akzeptiert.
- Bei Verbindungen aus anderen Netzen gilt immer grundsätzlich das Passwort, dass unter ALTER TTYPW gesetzt wurde.
- Der Sysop hat die Möglichkeit, durch Definieren eines GUESTCALL und Setzen der Variable HTTPGUESTFIRST auf den Wert 1 in *init.bcm*, den Login-Vorgang dahin abzuändern, dass generell bei HTTP-Zugriff der Login unter dem GUESTCALL erfolgt, wobei hier dann nur Lesezugriff gegeben ist. Das Passwort für den Gastzugang kann man mit SETUSER <guestcall> TTYPW <password> setzen. Siehe hierzu auch das Kapitel "19.9 Guest access using TCPIP".
- Das händische Generieren eines Passwortes (SETUSER <call> TTYPW <password>) für jeden User durch den Sysop kann umgangen werden, indem die Variable HTTPACCOUNT in *init.bcm* auf den Wert 1 gesetzt wird. Dann hat ein User die Möglichkeit sich unter dem GUESTCALL einzuloggen, und sich für sein Rufzeichen selber ein Passwort mit ALTER TTYPW zu setzen. Hierbei verbirgt sich jedoch das Risiko, dass jemand einem anderen ein Passwort setzt, und diesen dann aussperrt. Daher sollte diese Funktion eigentlich nicht angewendet werden! Ich rate vom Setzen von HTTPACCOUNT auf den Wert 1 also dringend ab!

Nach einem erfolgreichen Login setzt der HTTP Server beim Benutzer ein Cookie. Das Cookie dient dazu, bei einem späteren Login den Benutzer zu identifizieren und dadurch die Passwortabfrage zu umgehen. Wird dem Browser verboten Cookies zu setzen, dann muss das Passwort jedes Mal auf das Neue eingetippt werden. Als Browser eignen sich in jedem Fall die Produkte von Netscape bzw. Microsoft, alle anderen Browsern müssen Frames unterstützen. Die Funktionen der OpenBCM-Mailbox lassen sich komplett Fernbedienen, jedoch gibt es einige Ausnahmen:

- Beim Senden von Nachrichten muss immer die Referenz zu send angeklickt werden. Es hilft nichts, in der Befehlszeile einen Send-Befehl zu schreiben und dann auf Execute zu klicken.
- Die Bedienung von interaktiven Run-Utilities ist nicht möglich.
- Beim Empfang von Nachrichten werden Binärmails automatisch umgesetzt. Dabei wird der Mimetype beachtet und folgerichtig werden beispielsweise Mails mit einem binär eingespielten JPG-Bild automatisch angezeigt.

Alle anderen Funktionen sollten sich wie gewohnt bedienen lassen.

Die Oberfläche der Mailbox selber kann man als Sysop über den Befehl HTTPCSS konfigurieren:

- httpcss 0 einfache Oberfläche mit Frames und ohne CSS-Support
- httpcss 1 schicke Oberfläche ohne Frames und mit CSS-Support
- httpcss 2 einfache Oberfläche ohne Frames und ohne CSS-Support

Hier sollte man in der Regel den voreingestellten Wert auf 1 belassen, da man dann durch CSS die Möglichkeit hat, durch Editieren der Datei */bcm/http/style.css* mit einem Texteditor, die Oberfläche nochmals weiter anzupassen. CSS steht übrigens für "Cascading StyleSheets" und ist im Internet ein gängiges Mittel um Homepageseiten attraktiv und individuell zu gestalten. Wird der HTTP-Zugang allerdings nur über einen langsamen (schmalbandigen) Funkzugang genutzt, empfiehlt sich evtl. doch der Wert 0, um das Datenaufkommen pro Seitenaufruf zu minimieren.

In dem Verzeichnis */bcm/http* lassen sich übrigens weitere Dateien ablegen, die dann auch über HTTP abrufbar sind. Allerdings funktionieren nur die Endungen *.htm* (bzw. *.html*), *.gif*, *.wav* und *.jpg*.

Wird in */bcm/http* eine Datei *back.jpg* abgelegt, so wird diese als gekacheltes Hintergrundbild eingeblendet, ist eine Datei *qsl.wav* dort abgelegt, so wird diese bei jedem Aufruf ausgegeben. Eine Graphikdatei *logo.gif* wird ausgegeben, falls vorhanden. Die Datei *banner.gif* wird nur bei der CSS-Oberfläche genutzt und stellt, falls sie vorhanden ist, eine Bannergraphik am oberen Rand der Weboberfläche da. Ebenso wird die Datei *style.css* nur bei der CSS-Oberfläche benutzt und dient dazu, Farben, Fonts und Graphiken dem eigenen Geschmack anzupassen.

19.4. SMTP and POP3 access using Linux and Windows

Die OpenBCM-Mailbox unterstützt ein TCP/IP taugliches Mailinterface.

Das bedeutet: Emails können via SMTP in das Packet Radio Netz eingespielt werden, dabei werden die Mails automatisch umgesetzt und an die Packet Radio Verhältnisse angepasst. Auch die umgekehrte Richtung, nämlich das Abrufen der eigenen Mails via POP3 ist möglich. Auch hier erfolgt eine automatische Anpassung der Packet Radio Mails an das übliche Emailformat.

- Der OpenBCM-Mailbox-SMTP-Server ist auf Port 8025 erreichbar. Andere Werte lassen sich über *init.bcm* (Parameter SMTP_PORT) setzen. Die Erreichbarkeit über SMTP kann auch durch Setzen dieses Wertes auf 0 ganz abgeschaltet werden. Beim Versenden von Nachrichten ist es wichtig, die Absenderadresse im Mailprogramm korrekt einzutragen, also z.B. "dg9mhz@db0aab.#bay.deu.eu". Als mögliche Adressaten kommen Benutzer mit gültigen Rufzeichen oder Rubriknamen in Frage. Vom SMTP Server werden pro Mail bis zu 100 verschiedene Empfänger unterstützt, somit ist SMTP bei der OpenBCM-Mailbox ideal für die Verteilung von Mailinglisten.
- Zum Abrufen der eigenen Mails dient der OpenBCM-Mailbox-POP3 Server auf Port 8110. Wie üblich lassen sich andere Werte in *init.bcm* setzen.

Das Abrufen von Mails erfordert die Angabe eines Benutzernamens und Passwortes. Hier gelten die gleichen Konventionen, wie beim HTTP-Server.

Durch die automatische Umsetzung der Packet Radio Mails in Emails, verlieren die Mails keinerlei Information. Binärdateien und 7Plus Einspielungen werden automatisch Base64 kodiert. Ebenso automatisch wird der Mimetype gesetzt. Über SMTP eingespielte Mails bleiben von diesen Algorithmen unberührt und werden dadurch transparent durchgereicht.

Ein Konfigurationsbeispiel für db0aab anhand des Netscape Communicator's V4.01:

Zunächst wird über den mitgelieferten 'User Profile Manager' ein neues Profil angelegt. Die nun folgenden Formulare werden korrekt ausgefüllt:

Im Feld Namen bitte den vollständigen Namen eingeben, z.B. "Deti Fliegl". Als Emailadresse ist die komplette Adresse einzugeben, also z.B. "dg9mhz@db0aab.#bay.deu.eu". Das Profil nennt man dann z.B. "OpenBCM".

Nun gibt man noch mal Namen und Emailadresse ein. In der Zeile SMTP Server trägt man ein "db0aab.ampr.org:8025". Der Mailserver Username ist in dem Beispiel "dg9mhz". Als Incoming Mailserver gibt man an "db0aab.ampr.org:8110".

Der Communicator wird neu gestartet und nun wählt man unter "Edit/Preferences" die Kategorie "Mail&Groups/Mail Server" aus. In der Zeile unter POP3 gibt es eine Checkbox zum Anklicken, neben der steht: "Leave Messages on server after retrieval". Diese sollte angeklickt sein, damit die Packet Radio Mails auch nach der Abholung über POP3 weiterhin in der OpenBCM-Mailbox verfügbar sind. Wer das nicht will, kann die Option auch ausgeschaltet lassen.

Hier endet das allgemeine Konfigurationsbeispiel. Generell gilt: Wenn der TCP/IP Server keinen Nameserver besitzt (DNS), dann müssen entweder jeweils die IP Adressen an Stelle der Domainnamen direkt eingetragen werden oder man trägt in der Datei *hosts* des Betriebssystems die IP-Adressen und Namen per Hand ein:

- Bei Windows findet sich die Datei unter *\windows\hosts*
- Bei Windows NT/2000/XP unter *\winnt\system32\drivers\etc\hosts*
- Bei Unix/Linux unter */etc/hosts*

19.5. NNTP access using Linux and Windows

Der NNTP Server der OpenBCM-Mailbox befindet sich momentan (Mai 2004) noch in einem Betastatus. Der NNTP-Server ist auf Port 8119 erreichbar. Andere Werte lassen sich über *init.bcm* mit dem Parameter NNTP_PORT setzen. Die Erreichbarkeit über NNTP kann auch durch Setzen dieses Wertes auf 0 ganz abgeschaltet werden.

Es sind noch nicht alle NNTP-Befehle implementiert. So kann man momentan zwar alle Mails auslesen und auch Textmails versenden, ein Versenden von Binärdateien ist momentan jedoch noch nicht möglich.

Es ist nicht auszuschließen, dass es mit dem einen oder anderen Mailclient noch Probleme gibt. Gut ausgetestet und funktionieren tut das Programm "Microsoft Outlook Express", das jedem Windows-Betriebssystem beigefügt ist. Auch mit dem Newsreader von Netscape oder dem Programm TheBat kann man Mails zumindest problemlos abrufen und lesen.

Ein Konfigurationsbeispiel für Outlook Express und NNTP bei db0fhn via Internet: Unter "Extras/Konten" ein neues Newsgroup-Konto einrichten unter dem Button "Hinzufügen". Als Name den eigenen Namen eingeben und als Emailadresse die Packet-Radio-Adresse, z. B. dh8ymb@db0fhn.#bay.deu.eu. Als Serveradresse ist die IP-Nummer bzw. der Domainname des NNTP-Servers anzugeben, im Beispiel von db0fhn ist das "db0fhn.efi.fh-nuernberg.de". Die Option "Anmeldung am Newsserver

erforderlich" anklicken, da man in der Regel auch Mails versenden will. Bei Kontoname ist das eigene Rufzeichen und als Passwort das eigene Passwort (ALTER TTYPW bei Internetzugriff, im Hamweb der eigene Vorname) einzugeben. Damit ist der Account in Outlook Express eingerichtet und man kann sich nun alle verfügbaren Rubriken als Newsgroup herunterladen. Dann wählt man die Rubriken aus, für die man sich interessiert und "aboniert" diese. Durch Klick auf "Senden/Empfangen" sollten dann die gewählten Rubriken abgerufen werden.

19.6. FTP access using Linux and Windows

Der FTP Server der OpenBCM-Mailbox ist die jüngste Servererweiterung, scheint aber schon recht stabil zu laufen. Der FTP-Server ist auf Port 8023 erreichbar. Andere Werte lassen sich über *init.bcm* mit dem Parameter FTP_PORT setzen. Die Erreichbarkeit über FTP kann auch durch Setzen dieses Wertes auf 0 ganz abgeschaltet werden.

Alle Dateien, die im Filesurf angeboten werden, sind über den FTP ebenfalls downloadbar - es gilt hier also auch die Einstellung von FSPATH in *init.bcm*. Ein Upload ist nur in die bei FSPATH mit Schreibzugriff festgelegten Verzeichnisse möglich. Ein Löschen oder Umbenennen von Dateien ist generell nicht möglich.

Auf Userseite kann man z. B. das FTP-Programm "WS-FTP" verwenden.

19.7. Access to the service interface using Linux and Windows

Es existiert ein simples Service-Interface außerhalb der normalen Box-Oberfläche, das über Telnet erreichbar ist und auch darüber bedient werden kann.

Hier sind insgesamt vier Befehle möglich:

- TRACE ON/OFF — Startet/Stoppt die Ausgabe von *trace/syslog_r.bcm*
- MONITOR ON/OFF — Monitorfunktion an/aus
- CMD <Befehl> — Führt einen Boxbefehl aus
- QUIT — Verlässt das Service-Interface

Das Interface funktioniert nur dann, wenn der Host, auf dem das Telnet gestartet wird, in der Datei *rhosts.bcm* eingetragen ist und der Serv-Port entsprechend definiert wurde (normalerweise Port 8123).

19.8. Net-CMD interface using Linux

Die OpenBCM bietet ein Radio-Inferface (Net-CMD-Interface) unter Linux, wie es Wampes bzw. das Terminalprogramm TNT unterstützt, über dessen TCP/IP-Socket man Verbindungen nach außen aufbauen kann. Dazu muss zunächst mit dem Befehl RADIO_PORT die Portnummer des Radio-Interface definiert werden (standardmäßig auf 0 gesetzt, und damit deaktiviert).

Beispiel:
Radio-Port setzen:

```
radio_port 8134
```

Eine Telnet-Verbindung aus dem System mit

```
telnet localhost 8134
```

verbindet mit dem Radio-Interface, dort kann dann mit dem Befehl

```
CONNECT AX25 OE1XLR OE3DZW-12
```

als "oe3dzw-12" nach "oe1xlr" eine Verbindung aufgebaut werden.

Allerdings unterscheiden sich die Interfaces von Wampes und TNT, die Implementierung hat sich bei OpenBCM an TNT orientiert. Das Net-CMD-Interface ist aber kompatibel mit Wampes.

Nach einem Connect zu dem Socket, ist das Interface im Kommandomodus und akzeptiert drei Befehle: ASCII, BINARY und CONNECT. Eine Eingabe von anderen Befehlen oder falschen Parameter zu den drei Befehlen führt zu einem Beenden der Verbindung.

ASCII wählt eine Zeichenumwandlung von LF nach CR vor dem Senden von Daten auf der AX25-Seite und umgekehrt aus. Dies ist die Normaleinstellung.

BINARY wählt eine transparente Verbindung ohne Zeichenumwandlung aus.

CONNECT startet einen AX25-Verbindungaufbau. Es werden weitere Parameter benötigt, der genaue Syntax ist:

```
CONNECT <transport mode> <destination callsign> [source callsign]
```

Der einzig gültige Wert für <transport mode> ist AX25, andere Modi führen zu einem Beenden der Verbindung. Das <destination callsign> darf keine Digipeater beinhalten.

Nach einem erfolgreichen Verbindungsaufbau wechselt das Interface in den Datamodus: alle empfangene Daten werden an das Socket gesendet, alle Daten vom Socket werden auf der AX25-Seite übertragen.

Wenn der Verbindungsaufbau nicht erfolgreich war, wird die Interfaceverbindung ohne weitere Infos beendet.

Beim Verbindungsaufbau werden die Pfade entsprechend einer Datei *netpath.bcm* verwendet:

```
------ l2path.bcm
; comments are ignored (start with ;)
; format: <to_call>: <path>
db0clx:  db0clx oe1xlr
ha5dxx:  ha5dxc oe1xlr oe3xpr
------
```

19.9 Guest access using TCPIP

Der Sysop hat die Möglichkeit, durch Definieren eines GUESTCALL in *init.bcm* ein Gastrufzeichen festzulegen (in der Regel ist dies "GUEST"). Beim Einloggen mit dem Gastrufzeichen ist kein Schreibzugriff auf die Mailbox erlaubt. Man kann dies also nutzen, um z. B. Internet-Benutzern, die unter Umständen ja nicht lizensiert sind, einen reinen Lesezugriff auf die Mailbox zu gewähren. Als Sysop gibt man also

```
guestcall GUEST
```

ein.

Damit der Gastzugang unter TCPIP nutzbar wird, muss nun noch ein Passwort für den Gastzugang gesetzt werden. Dieses Passwort kann man als Sysop mit dem Befehl SETUSER <guestcall> TTYPW <password> definieren. Man gibt nun also noch z. B.

```
setu GUEST ttypw test
```

ein.

Beim Login mit Gastrufzeichen hat man ferner auch nicht die Möglichkeit ALTER-Einstellungen zu diesem Rufzeichen zu setzen bzw. zu ändern, damit nicht ein Gast aus Unwissenheit den Namen bzw. die Heimatmailbox dieses Gastrufzeichens verstellt. Diese Einstellungen können daher nur vom Sysop per SETUSER Befehl vorgenommen werden.

Der Gastzugang kann sowohl unter HTTP, NNTP, FTP als auch Telnet sinnvoll genutzt werden.

Ferner ist es durch das zusätzliche Setzen der Variable HTTPGUESTFIRST auf den Wert 1 in *init.bcm* möglich, das Login-Verhalten bei HTTP-Zugriff dahin abzuändern, dass generell bei HTTP-Zugriff auf die Mailbox zunächst der Login

automatisch unter dem GUESTCALL erfolgt. Ein HTTP-Benutzer muß sich dann bei Schreibzugriff per Klick auf den USERLOGIN Link per Login und Passwort identifizieren. Der Sysop sollte also auch noch

```
httpguestfirst 1
```

eingeben.

20. Multilanguage function of OpenBCM

Die OpenBCM-Mailbox in der Lage, bis zu 40 verschiedene Sprachen zu sprechen, derzeit sind folgende Sprachen vorhanden:

- DL: Deutsch
- DLA: Deutsch mit ANSI-Sequenzen (zur Farbsteuerung)
- BAD: Badisch
- BAY: Bayerisch
- BW: Schwäbisch
- CT: Portugiesisch
- EA: Spanisch
- FF: Französisch
- GB: Englisch
- HA: Ungarisch
- HRV: Kroatisch
- I: Italienisch
- JA: Japanisch
- KL: Kölsch
- LX: Letzeburgisch
- NL: Holländisch
- OK: Tschechisch
- OM: Slowakisch
- PF: Pfälzisch
- PL: Polnisch
- RUS: Russisch
- S5: Slowenisch
- TA: Türkisch
- TRK: Türkisch mit Akronymen

Aktuell werden immer nur die Sprachen DL und GB gepflegt, alle anderen Sprachen können unter Umständen einem älteren Stand entsprechen, da es nicht immer fleissige Übersetzer für diese Sprachen gibt.

Hinweise:

- Alle Sysopmeldungen sind Englisch. Um den Aufwand vertretbar zu halten, sind die Meldungen am Bildschirm und die für die reinen Sysop-Befehle nicht änderbar und permanent in englischer Sprache eingebaut. Da es sich aber meist um einfache Ausdrücke handelt, dürften keine Probleme zu erwarten sein.
- Änderungen an der Boxsoftware, insbesondere Erweiterungen der Funktionalität bedingen gelegentlich eine Erweiterung der Dateien mit den Meldungen. Das erschwert den Updatevorgang erheblich, besonders wenn sehr viele Sprachen existieren. Seit der ersten Stunde der Mehrsprachigkeit ist bis jetzt das Format der messages-Datei mehrfach angepasst worden. Es ist vorherzusehen, dass in einer der nächsten Versionen nochmals ein Update erforderlich wird, weil schon recht viele Meldungen in der Box festverdrahtet in englischer Sprache eingebaut sind.
- Die Meldungs-Dateien müssen im Format exakt stimmen. Eine Zeile zu viel oder zu wenig kann unvorhersehbare Folgen haben. Eine gewissenhafte Pflege dieser Dateien ist deshalb wichtig.

- Wird eine Datei zu einer angewählten Sprache nicht gefunden, so wird zunächst nach einem englischen Text (Endung *.GB*) gesucht, und wenn auch dieser nicht vorhanden ist wird die Meldung in Deutsch (*.DL*) ausgegeben. Einer der beiden Meldungsblöcke *.GB* oder *.DL* muss existieren, sonst erfolgt ein Abbruch der Box. Analoges gilt für die Dateien INFO, AKTUELL, HELP, QTEXT, etc., allerdings ist deren Existenz nicht lebenswichtig.

Erzeugung einer neuen Sprache:
Um eine neue Sprache zu erzeugen, muss in erster Linie eine Datei *msg/messages.<Sprachkenner>* erzeugt werden. Der <Sprachkenner> ist dabei frei wählbar, sollte aber möglichst den ersten zwei Buchstaben des Calls eines Landes in dem diese Sprache gesprochen wird entsprechen. Vordefiniert sind *.DL* für Deutsch und *.GB* für Englisch. Es sind die Format-Konventionen für eine Sprachdatei (siehe auch 20.1. Syntax of a language file) genau zu beachten! Anschließend muß die Datei *speech.bcm* um eine neue Zeile erweitert werden.

20.1. Syntax of a language file

Die erste Zeile einer Sprach-Datei enthält eine Beschreibung der Sprache und eine Versionsnummer. Beides kann man beim Aufruf von "A S" (ohne weitere Parameter) abrufen.

Die Landeskenner werden in der Datei *speech.bcm* verwaltet, siehe Kapitel "5.8. Multilanguage configuration (speech.bcm)".

Ab der zweiten Zeile kommen die eigentlichen Meldungen. Sie sind in einer C-kompatiblen Form abgelegt. Zu beachten ist folgendes:

- Jede Zeile beginnt und endet mit einem Anführungs-Zeichen. Beginnt eine Zeile mit ";", so wird sie ignoriert. Zeichen nach dem zweiten Anführungszeichen werden ebenfalls ignoriert.
- Steuerzeichen werden in C-Syntax abgelegt. Ausgewertet werden
  - \n: Return (Zeilenende)
  - \a: CTRL-G (Klingelzeichen)
  - \": Anführungszeichen
  - \\: Backslash
- Variablen, die innerhalb von den Ausgabestrings ausgegeben werden, werden als C-Formatstring abgelegt. Folgende Symbole tauchen auf:
  - %s: String
  - %c: Einzelzeichen
  - %d, %u, %ld: Dezimalzahl

  Dabei darf die Position der Ausgabefelder verändert werden, keinesfalls aber der Typ oder die Reihenfolge.
- Es gibt Ausgaben, deren Länge exakt stimmen muss (z.B. Überschriften). Dies ist meist aus dem Zusammenhang zu ersehen und sollte beachtet werden.
- Die beste Methode beim Übersetzen ist, die deutsche oder englische Datei umzubenennen, in den Editor zu nehmen und Zeile für Zeile zu übersetzen.

Die Indizes für die verschiedenen Help-Dateien (*msg/help.**) werden auf der Platte verwaltet. Dadurch entstehen zusätzliche Dateien (*msg/helpidx.**). Diese werden bei Änderung der Help-Datei automatisch angepasst, es ergibt sich kein Handlungsbedarf für den Sysop.

Es können maximal 40 Sprachen verwaltet werden.

20.2. Syntax of a help file

Neben der Datei *msg/messages.** müssen auch die anderen Text-Dateien, insbesondere *msg/help.** übersetzt werden. Die Endung dieser Dateien muss dann passend zum jeweiligen Sprachkenner sein.

Bei der Datei *msg/help.<Sprachkenner>* werden Makros nicht expandiert, ein "%s" wird als solches ausgegeben. Die Help-Datei besteht aus durch Haupt- und Unterbegriffe referenzierte Texte. Die Begriffe werden mit "\\begriff" in der ersten Spalte in einer eigenen Zeile definiert.

Ein "help alter forward" kann mit "h a f" abgekürzt werden und gibt einen anderen Helptext als bei Eingabe von "help alter" aus. In der Help-Datei werden solche Kombinationen mit einem Punkt abgetrennt.

Beispiel:

```
\\ALTER
--- Hilfe-Text für den Befehl ALTER ---
\\ALTER.FORWARD
--- Hilfe-Text für den Befehl ALTER FORWARD ---
```

Es ist nur eine Hierarchieebene möglich, Konstrukte wie "\\DIR.USER.MESSAGES" funktionieren nicht.

Außerdem können für redundante Helptexte Verknüpfungen angelegt werden. Dies erfolgt mit dem Zeichen "=".

Beispiel:

```
\\MSG=TALK
--- Kein Text! ---
```

ruft bei der Eingabe "help msg" den Helptext von "talk" auf.

```
\\MYBBS=ALTER.FORWARD
--- Kein Text! ---
```

ruft bei Eingabe von "h mybbs" den Helptext von "alter forward" auf.

Auch die umgekehrte Verknüpfung wäre zulässig:

```
\\ALTER.FORWARD=MYBBS
--- Kein Text! ---
```

Es lassen sich also Unter- und Überbegriffe beliebig vermischen. Verknüpfungen dürfen wiederum auf Verknüpfungen zeigen, zyklische Verknüpfungen sind natürlich unzulässig und führen zu einer Fehlermeldung im Syslog.

Am Anfang der Help-Dateien steht meist der Hilfebegriff \\HVERSION. Es hat sich eingebürgert unter diesem Begriff eine Versionsnummer mit Datum und Ersteller abzulegen, um unterschiedliche Stände kenntlich zu machen. Als Benutzer kann man dann mit "H HVER" diese Versionskennung leicht abfragen.

Ich bitte dringend darum, jede Übersetzung an BAYCOM @ BABOX zu senden, um eine zentrale Lagerhaltung aller Texte zu ermöglichen.

21. One-Letter boards

One-Letter boards can be only read/listed with sysop rights. Some one-letter boards have a special meaning:

Board	**Software**	**Used for**
B	DieBox, OpenBCM	Erased mails
C	DieBox	Erased mails via KILL
E	DieBox, OpenBCM	Remote erase information exchange (obsolete)
F	-	Internal sysop

		information distribution
M	DieBox, OpenBCM	MYBBS information exchange (obsolete)
P	Pocsag, OpenBCM	Used for POCSAG messages
R	-	DieBox software distribution
T	DP-Box, DieBox, OpenBCM	Used by TELL command
V	DP-Box	DP-Box software distribution
W	DP-Box, OpenBCM	WPROT information
X	DP-Box, WinGT	Forwardqueue
Z	-	Clipboard for mails

One-Letter boards can't sometimes not be used with the DIR command, because they are hidden by a DIR subcommand (e.g. "DIR A"=DIR ALTER, "DIR B"=DIR BOARDS, "DIR N"=DIR NEWS, etc.). These boards can be reached with the LIST command or if you change the board before with CD command and than use the DIR command without the boardname.

22. Upgrade DOS version to Windows or Linux version

22.1. Upgrade from DOS/Windows to Linux

Der Umstieg von der DOS- oder Windows-Version auf Linux ist unkomplizierter, als man vermuten möchte! Alle vorhandenen Mails können 1:1 auf das Linux-System kopiert werden und gehen bei der Umstellung nicht verloren!

Es sind jedoch ein paar Kleinigkeiten zu beachten, die nachfolgend beschrieben werden:

Es ist vor der Umstellung zunächst zu überprüfen, welche BCM/OpenBCM-Version unter DOS/Windows eingesetzt wird.

Ab OpenBCM v1.04 verwendet die Mailbox die Userdatenbank *users4.bcm*, die auch unter Linux verwendet wird. Diese kann also unter Linux 1:1 übernommen werden, der nachfolgende Absatz kann dann also übersprungen werden.

Wird unter DOS/Windows eine ältere OpenBCM-Version oder gar noch eine BCM-Version eingesetzt, die noch die Userdatenbank *users.bcm* einsetzt, so muss diese manuell mit einem Konvertiertool umgestellt werden oder zunächst die DOS/Windows-Version auf den aktuellsten Stand gebracht werden, wobei dann die Konvertierung automatisch geschieht. Entscheidet man sich für die manuelle Konvertierung, kann diese nur unter DOS bzw. unter Windows im DOS-Fenster erfolgen, jedoch nicht unter Linux! Man benötigt für die manuelle Konvertierung das Programm *cvusers4.exe*, das einfach im DOS-BCM/OpenBCM-Verzeichnis zu starten ist. Nachdem *cvusers4.exe* erfolgreich durchgelaufen ist, sollte eine Datei *users4.bcm* entstanden sein - die Userdatenbank für die aktuelle OpenBCM-Version.

Als nächstes kopiert man nun die Datei *users4.bcm* und die Dateien *userh2.bcm*, *hadr4.bcm* und *hadrhash.bcm* auf das Linux-System ins dortige OpenBCM-Verzeichnis. Dabei ist darauf zu achten, dass die Dateinamen komplett aus Kleinbuchstaben bestehen, ggf. ist also mit dem Shell-Befehl "mv" der Dateiname unter Linux anzupassen!

Die BID-Datei ist nicht konvertierbar und muss unter Linux durch REORG F aus dem Mail-Datenbestand neu erzeugt werden.

Der nächste Schritt ist dann die Überführung aller vorhandenen Daten (also komplettes OpenBCM-Verzeichnis sowie USER und INFO-Verzeichnisbäume) von DOS/Windows nach Linux. Folgende Möglichkeiten sind denkbar:

- Benutzung eines neuen Rechners und Übertragung der Dateien von alt nach neu über Ethernet (mit NFS, Samba-SMB oder FTP)
- Einbau einer neuen Platte, Installation von Linux, mounten der alten Platte als MSDOS-Dateisystem, dann umkopieren der Daten auf die neue Platte
- Sicherung aller Dateien auf ein vorhandenes Medium, das sowohl unter DOS/Windows als auch unter Linux ansprechbar ist (z.B. CD-ROM), dann Einspielen auf die unter Linux neu formatierte Platte

Das OpenBCM-Home Verzeichnis sollte unter */bcm* angelegt werden. Hier werden alle Dateien der DOS/Windows-Version mit allen Unterverzeichnissen hineinkopiert. Selbstverständlich ist es sinnvoll, hierbei die rein DOS/Windows-spezifischen Dateien auszumisten. Die Verzeichnisse für die *user*- und *info*-Daten können in *init.bcm* frei gewählt werden, sind aber im Regelfall Unterverzeichnisse vom Hauptverzeichnis */bcm*.

Werden mehrere Platten verwendet, so müssen hier sinnvolle Mountpunkte definiert werden. Es kann nicht im Rootverzeichnis von Platten oder Partitionen gearbeitet werden, da hier Linux das lost+found-Verzeichnis anlegt, was die Mailbox durcheinander bringen kann. Es ist also notwendig, vom Mountpunkt stets ein Unterverzeichnis zu definieren. Die Pfade sollten jedoch nicht zu lang sein, da dies etwas die Performance mindert.

Sind alle Dateien umkopiert und die Pfade eingestellt, so kann die Box gestartet werden. Vor einem Zugriff über Funk sollten noch nacheinander "REORG L", "REORG I", "REORG H" und "REORG F" gemacht werden. Hiermit wird auch die BID-Datei neu aus den Mail-Datenbeständen erzeugt. Hinweis: Jeder REORG muss abgewartet werden, ehe der nächste gestartet werden kann.

Die beiden Versionen unter DOS/Windows und Linux verhalten sich bezüglich Dateisystem so identisch, wie nur möglich. Ein Fallstrick ist die Tatsache, dass bei UNIX/Linux Dateinamen üblicherweise kleingeschrieben werden, bei DOS ist GROSS-/Kleinschreibung egal und es wird dummerweise meist alles GROSS geschrieben. Das schafft Verwirrung und war auch in der Software nicht ganz einfach zu implementieren, zumal viele Dateinamen aus dem Zusammenhang automatisch erzeugt werden. Die Linux OpenBCM-Mailbox verwendet ausschließlich kleingeschriebene Dateinamen, egal wo sie auftauchen.

Es darf (fast) überall die Folge CR/LF statt der UNIX/Linux-üblichen Folge LF in den Dateien vorkommen. Umgekehrt kommt auch die DOS-Version mit nur-LF aus. Ausnahme sind *list.bcm* und *check.bcm* Dateien. Diese müssen bei beiden Versionen stets mit CR/LF abgeschlossen sein. Diese werden jedoch üblicherweise nicht mit einem Editor bearbeitet und sind daher unproblematisch.

Wichtig ist, dass nach der Installation alle Dateien (auch das Verzeichnis selbst!) dem User "bcm" gehören (siehe oben). Sicherheitshalber sollte also:

```
cd /bcm (Home-Verzeichnis der OpenBCM)
cd ..
chown -R bcm:bcm bcm
chmod -R u+rw bcm
```

aufgerufen werden.

22.2. Upgrade from DOS to Windows

Spätestens mit der Verbreitung von Windows XP (vorher Windows NT und 2000, Windows 3.1/95/98/ME ist jedoch unbrauchbar) ist quasi fast jedem ein auch für den Mailboxbetrieb durchaus nutzbares Betriebssystem bekannt, dass im Vergleich zu DOS vor allem das lästige 640 kB Hauptspeicher-Problem verbannt und ein vernünftiges Dateisystem NTFS zur Verfügung stellt. Schon alleine dies sind gute Gründe von der DOS-Version wegzukommen. Sicherlich ist die Linux-Welt eine noch

bessere Wahl, da es hier vor allem viel mehr Packet-Radio-Software gibt als unter Windows.

Die Windows-Version verhält sich sehr viel ähnlicher zur Linux- als zur DOS-Version. Das ist auch kein Wunder, da im 32 Bit-Mode vieles aus der DOS-Welt (z.B. Hardware- und BIOS-Zugriffe) nicht zur Verfügung steht, und auch z.B. die Speicherverwaltung eher vergleichbar mit Linux ist.

Der Unterschied zu DOS ist:

- Es ist keine direkte Benutzeroberfläche vorhanden. Ein Login ist nur über Telnet (Standard: TCP-Port 4719) und HTTP (Standard: TCP-Port 8080) möglich.
- Der Layer2 für Packet-Radio ist in *bcm32.exe* enthalten. Es ist allerdings nur KISS und AXIP über UDP möglich. Die Parametrierung erfolgt über *init.l2*, wie bei Linux.

Alle Datenstrukturen sind kompatibel zur DOS-Version. Es muss hier also beim Umstieg von DOS zu Windows nichts konvertiert werden.

23. Upgrade from BCM v1.42n to OpenBCM

Die Version 1.42n war die letzte BCM-Version, die von OE3DZW herausgegeben wurde. Einige Sysops sind dann erst mal auf diesem Stand stehen geblieben und wollten abwarten, wie sich die Software weiterentwickelt. Mittlerweile sind viele Neuerungen in die Software eingeflossen und es kommt immer wieder die Frage auf, was sich denn bislang Grundlegendes seit der Version 1.42n geändert hat. Dieses Kapitel soll hierüber Klarheit schaffen und somit helfen von der mittlerweile doch recht alten BCM Version 1.42n endlich wegzukommen!

Bei der Umstellung einer alten Baybox v1.42n auf die aktuelle Version der OpenBCM ist prinzipiell wenig zu beachten.

Alle Mails und bisherigen Konfigurationseinstellungen bleiben unverändert erhalten!

GANZ WICHTIG: beim Umstieg von BCM 1.42n auf die aktuelle OpenBCM v1.05 sind unbedingt alle alten *msg/messages.** Dateien im Verzeichnis *msg* zu löschen, da diese die neue Version zum Absturz bringen können!

Aktuelle *msg/messages.** Dateien sind im Filesurf von DB0FHN (Amateurfunk), im Filesurf von DB0274 (CB-Funk) oder im Internet auf der Webseite http://dnx274.dyndns.org/baybox zu bekommen. In der aktuellen *msg_*.zip* Datei, die man an o. g. Stellen bekommen kann, sind auch aktuelle *msg/help.** Dateien enthalten, die die Onlinehilfe der Mailbox auf den aktuellen Stand bringen.

Nun ein Überblick über die wichtigsten Änderungen:

- **Maileingabe:**
  - Abbruch einer Mail anstatt nur mit CTRL+X nun auch mit /AB oder /ab möglich
  - Beim Absenden von Mails wird nun wie bei der DPBox auch der spätere Verteilweg angezeigt
  - Reply: Es wird nun auch der neue Titel angezeigt
  - Lifetime-Query wenn LTQUERY in *init.bcm* auf 1 oder 2 gesetzt

- **Mailforward:**
  - Dateiforward funktioniert nun problemlos (Befehle FWDIMPORT bzw. FWDEXPORT)

- Erweiterung File-Forward: wird in *fwd.bcm* als Connectpfad das Schlüsselwort "FILE:" angegeben, werden Mails automatisch im-/exportiert
- Handling von Forward-Queue-Dateien verbessert
- Box kann als reine Forward-Box betrieben werden (Befehl: SFONLY)
- ausgehender TELNET-Forward nun auch unter Windows möglich
- Ist eine der Forward-Partner der eigenen Box eine DieBox, so ist bei dieser DieBox in der Datei *mbsys\sfwid.box* folgender Eintrag einzufügen, damit AUTOBIN-Mails richtig übertragen werden:

```
BayCom-1.1  18 S
BayCom-1.2  18 S
BayCom-     19
OpenBCM-    19
```

- Autorouter integriert (Befehle AUTOFWDTIME und AFWDLIST) als Ergänzung zu den statischen *fwd.bcm*-Definitionen
- in *fwd.bcm* gibt es neue Optionen bzgl. 7+ und Binärdatei-Unterdrückung

- **User-Datenbank erweitert und neue ALTER Optionen für jeden User**:
  - mit ALTER PS kann man die Optionen für den PS Befehl festlegen
  - mit ALTER PACLEN kann jeder User seine Paclen festlegen
  - ALTER AWAY Funktion neu hinzu
  - ALTER ZIP und ALTER QTH neu hinzu
  - ALTER NOTIFICATION Funktion neu hinzu
  - ALTER FBBCHECKMODE für FBB-Kompatible Checklisten hinzu
  - ALTER REJECT kann auch auf Absender gesetzt werden

- **Ausgaben optimiert:**
  - PS-Ausgabe
  - ST F-Ausgabe zählt übertragene Mails, Statusanzeige verbessert
  - PATH-Ausgabe kommt mit langen H-Path zurecht, zeigt auch WPROT-Routing an
  - VERSION zeigt Filesurf-Speicherplatz an, Anzeige aller eincompilierten Features

- **Zahlreiche neue Funktionen in *init.bcm*:**
  (vgl. hierzu auch das Kapitel "24.4. Initialization *file* init.bcm")

- callformat	zum Umstellen von CB/Amateurfunkversion
- mailbeacon	Konfiguration der Mailbake
- nopopsmtp	POP3-Mailzugriff ohne vorherigen SMTP-Zugriff
- httpguestfirst	Bei HTTP-Zugriff zunächst immer GAST-Status
- httpaccount	User können selber ein HTTP-Account anlegen
- httpttypw	TTYPW Passwort auch für das AMPRNET verwenden
- defreadlock	Default-Einstellung für ALTER READLOCK
- deffbbcheckread	Default-Einstellung für ALTER FBBCHECKREAD
- ltquery	Lifetime-Abfrage aktivieren/deaktivieren
- unsecurettypw	Konfiguriert den Zugriff auf ALTER TTYPW
- maillistsender	Konfiguriert das Absendercall vom Mailserver
- summertime	Konfiguriert Sommer-/Winterzeit (Windows)
- altboardinfo	Board-Info aus *boardinf.bcm* mit/ohne Return
- timeoutwarning	Timeout-Warnung vor Timeout-Disconnect
- asklogin	Abfrage von Userdaten bei neuen Usern
- tellmode	Konfiguriert/Aktiviert den Tell-Server
- usrtimeout	Legt ein User-Timeout fest
- sfonly	Aktiviert/Deaktiviert reine S&F-Mailbox
- smoothheader	Kompletten oder verkürzten Mailheader ausgeben
- nounknownroute	Mails mit unbekannter Route ablehnen
- addlinuxsystemuser	Support für Linux-User-Password

- **Didadit-Unterstützung:**
  (Didadit-fähige Terminalprogramme sind LinKT, Paxon, WPP und Winstop)

- Didadit-Transfer-Protokoll Spezifikation v0.91 von DG4IAD/DH3MB implementiert
- Didadit-Transfer-Protokoll wird nun im Filesurf unterstützt (Up/Download)

- **CB-Funk**:
  - CB-BCMNET Loginkonzept und Erweiterungen implementiert
  - es werden mehr CB-Rufzeichenmuster erkannt

- **TCPIP-Funktionen:**
  - NNTP-Server erweitert
  - neuer FTP-Server auf Port 8021 bietet Filesurf-Zugriff per TCP/IP
  - SMTP File-Attachment werden akzeptiert und in AUTOBIN umgewandelt
  - HTTP-Server-Oberfläche verbessert (u.a. Befehlszeile, /bcm/http/qsl.wav, logo.gif, automatische Passworterstellung für Internet-Boxen...)

- **PW & Hold/Reject:**
  Seit dem Jahr 2000 ist die OpenBCM mit einem erweitertem PW & HOLD/REJECT-System von DF3VI ausgestattet, doch leider wird dies bislang so gut wie gar nicht angewendet. PW & HOLD/REJECT ist ein zweiteiliges System, mit dem sich die Sicherheit der Benutzer gegen Rufzeichenmissbrauch bedeutend erhöhen lässt:
  - Schutz für Benutzer, die noch kein Passwort benutzen
  - Hold-Mechanismus für Nachrichten aus ungeschützter Quelle

  a) Passwortverfahren

  Das geänderte Passwortverfahren wird mit dem Parameter "USERPW 2" aktiviert.
  Damit gelten automatisch eine Reihe Schutzmassnahmen für Benutzer ohne Passwort, was den Zugriff auf Nachrichten betrifft:
  - kein unauthorisiertes Löschen
  - kein Schreiben von (gefälschten) Nachrichten mit Forward
  - kein Lesen von fremden Usermails
  - kein Verstellen von wichtigen Parametern (wie MyBBS)
  - Aufruf von Runutils nur mit User-Passwort (Parameter "-P" in Datei *runutil.bcm* dient dazu, dass Programme nur von Benutzern mit User-Passwort aufgerufen werden können)

  In der Datei *msg\pwonly.<sprache>* kann man einen Hinweistext für Benutzer ohne Passwort hinterlegen, der beim Login gesendet wird. Hier sollte man insbesondere darauf hinweisen, wie man ein Passwort bekommt.
  Im Gegensatz zum bisherigen Verhalten, wo unter Rufzeichenmissbrauch jeder jedem ein Passwort setzen (und somit aussperren) kann, wird das Passwort nun vom Sysop eingerichtet. Mit den Befehlen PWGEN und SETPW lässt sich eine automatische Passwort-Vergabe durch den Sysop einrichten: Dazu wird mittels PWGEN eine Datei *userpw.bcm* mit 81 Passwörtern vorbereitet, die dann mit "setpw <user> <nr>" Usern zugeordnet werden können. Diese Datei *userpw.bcm* sollte man auch zu Hause haben, um das zugeteilte Passwort dem User mitteilen zu können (natürlich nicht über Funk auslesen!).
  Mit dem Parameter "USERPW 3" wird das User-Passwort nicht mehr beim Login, sondern erst später abgefragt - analog zur DP-Box. So wird das User-Passwort nur noch zum Schreiben und Löschen von Mails sowie zum Verstellen von Einstellungen benötigt. Vorteil: ein vergessenes Passwort sperrt einen nicht gleich aus.

  b) HOLD-Mechanismus

  Der HOLD/REJECT-Mechanismus wird über die Datei *reject.bcm* gesteuert. Zum bequemen Verwalten der Einstellungen gibt es den REJECTEDITOR. In *reject.bcm* kann man parametrieren, wie mit Nachrichten verfahren wird, die ohne Passwort gesendet werden oder über ungeschützte S&F-Wege hereinkommen. Mindestens ein Eintrag "P .B" (kein Bulletin-Send ohne AX25-PW) sollte vorhanden sein. Alle Optionen, die in *reject.bcm* angewendet

werden können, sind im Kapitel "5.10. Reject and Hold (reject.bcm)" beschrieben.
Wie lange angehaltene Nachrichten nicht geforwarded werden, wird über den Parameter HOLDTIME in Stunden eingestellt (typisch: 48 Stunden = 2 Tage). Mit DIR HOLD lassen sich angehaltene Nachrichten aufzeigen. DIR HOLD funktioniert genauso wie DIR NEWS oder CHECK abhängig vom LastCheck-Timer. Mit "HOLD -u" bzw. "FORWARD -h" kann der Sysop auf Hold gesetzte Mails wieder in den Forward geben.

- **DOS:**
  - Nutzung eines externen Watchdogprogramms möglich
  - Telefonmodem-Support implementiert

- **Linux:**
  - Befehl "addlinuxsystemuser 1" aktiviert ALTER LINUXPW, wo sich User ein Userlogin im Linuxsystem setzen können
  - Befehl "lddversion" zeigt eincompilierte Libs unter Linux an

- **Diverses:**
  - M_Filter-Support implementiert
  - *trace/swaplog.bcm* protokolliert von anderen Mailboxen geswappte Mails
  - Unterscheidung DISC/KILL in SLOG
  - *speech.bcm* zur Verwaltung der Sprachdateien *msg/messages.**
  - Usertimeout durch USRTIMEOUT in *init.bcm* möglich
  - Autosysop-Funktion über *asysop.bcm* implementiert
  - WPROT Funktionen erweitert
  - Wartezeit beim Logout mit Ausgabe des QTEXT reduziert
  - Desaster bei der automatischen Umstellung zwischen Sommer-/Winterzeit gelöst
  - POCSAG-Server-Benachrichtung implementiert
  - QT-Befehl erweitert
  - *crontab.bcm* kann nun auch z.B. "*/5" für fünfminütige Aufrufe verarbeiten
  - HOLD Befehl erweitert und korrigiert
  - statt *convname.bcm* und *convlife.bcm* wird nur noch *convert.bcm* verwendet, wo eine weit bessere Einsortierung von Mails automatisiert werden kann, u. a. nach dem Titel der Mail. Der genaue Syntax von *convert.bcm* ist im Kapitel "5.11. Automatic board conversion (convert.bcm)" erklärt. Am Anfang kann man seine *convname.bcm* auch erst mal nur in *convert.bcm* umbenennen, die Funktion der alten Einträge bleibt damit erhalten.
  - interner *fwd.bcm* und *reject.bcm* Editor (Befehl FWDEDIT, REJEDIT)
  - neuer Befehl EXTRACT zum Extrahieren von Dateianhängen einer Mail auf die Mailboxfestplatte, das externes EXTRACT Tool von DF3VI ist damit nicht mehr nötig
  - *reject.bcm* um ODER-Verknüpfung erweitert
  - Passwort-Verfahren nach DPBox (Befehl USERPW)
  - Erweiterung des Conversmodus um Talk-Befehle
  - Tell-Funktion wie bei Diebox/DPBox implementiert
  - Ping-Funktion neu hinzu
  - neuer Befehl UNREAD
  - Boxintern wird alles in UTC-Zeit abgewickelt, für den User in Lokalzeit, dadurch u.a. Änderung von logfile Format, evtl. müssen deshalb ein paar alte Runutils angepasst werden (Statistik, ULOG/FLOG...)
  - LOG-Befehl überarbeitet, u. a. ist nun eine statische Zusammenfassung möglich
  - Bei interaktiven Runutils (d.h. Runutils die auf Usereingaben warten) muss nun in *runutil.bcm* die Option "-A" gesetzt werden

...und viele, viele weitere kleine Verbesserungen und Korrekturen!

In der Dokumentationsdatei *history.txt* ist die komplette Weiterentwicklung der Baybox BCM von v1.00 bis v1.46 dokumentiert. Seit OpenBCM liegt dem Quellcode die Datei *changes.txt* bei, in der alle Änderungen der OpenBCM ab v1.00 dokumentiert sind. Manchmal ist auch ein Blick in diese Dateien sehr interessant.

Zusammengefasst müssen also folgende Punkte beim Upgrade ausgeführt werden:

- Sprachdateien *msg/messages.** und Onlinehilfedateien *msg/help.** ersetzen
- neue Mailbox-Hauptdatei *bcm.exe*, *bcm32.exe* oder *bcm* ersetzt die alte Version
- ggf. *convname.bcm* in *convert.bcm* umbenennen
- die neue Mailbox-Version starten, dabei wird die Userdatenbank automatisch ins neue Format konvertiert
- ggf. die oben aufgelisteten neuen Funktionen konfigurieren und ausprobieren

24. Specification of system files

24.1. Target

Alle Dateien sind darauf ausgelegt, dass die Mailbox insgesamt den folgenden Anforderungen gerecht wird:

- Robuster, pflegeleichter Betrieb: Durch Betriebsausfälle und Fehlbedienungen dürfen keine Inkonsistenzen in den Daten (insbesondere beim Forwarding-Betrieb) entstehen. Die Dateistruktur muss zu jedem X-beliebigen Zeitpunkt immer restaurierbar bleiben.
- Konstante Transaktionszeit: Unabhängig von der Anzahl der Mails in der Mailbox darf die Verarbeitungsgeschwindigkeit nicht unter ein gegebenes Maß gehen. Gefordert sind:
  - max. 200 ms ununterbrochener Betrieb eines Prozesses
  - max. 1 Sekunde Antwortzeit bei allen einfachen Transaktionen
  - keine Blockierung des Systems durch komplexe Transaktionen (Suchvorgänge etc.) oder Hintergrundoperationen
- Alle Wartungsarbeiten am System müssen ohne Abstellen der Software und aus der Ferne möglich sein.

24.2. Structure of directories and files

Für die System-Dateien und -Verzeichnisse der OpenBCM-Mailbox gelten folgende Grundsätze:

- Alle Verwaltungsdateien haben die Erweiterung *.bcm*
- Die Mails in der Box werden voll auf die Dateistruktur abgebildet. Der dadurch entstehende Overhead wird in Kauf genommen.
- Jeder Benutzer, für den mindestens eine Nachricht vorhanden ist, erhält ein eigenes Verzeichnis.
- Jede Rubrik (Board) erhält ein eigenes Verzeichnis. Der Name dieses Verzeichnisses setzt sich aus maximal 8 Buchstaben zusammen, wobei nur der Dateiname benutzt wird. Die Dateinamen-Extension (nach dem Punkt) bleibt leer.
- Das Anlegen von Unterverzeichnissen darf beliebig geschachtelt werden. Es entsteht ein Verzeichnis-Baum, der voll im Bulletinsystem der Mailbox abgebildet wird. Beispiel:

```
MAILBOX--+--USER--+--DG3RBU
         !        !
         !        +--DL8MDW
         !        !
         !        +--DL8MBT
         !        !
         !        +--OE3DZW
         !
         +--INFO--+--TMP-------+ )
                  !            + ) Temporäre Verzeichnisse für alle
                  !            + ) Boardnamen, die nicht in der Stuktur
                  !            + ) vorhanden sind. Diese Verzeichnisse
                  !            + ) werden gelöscht, wenn sie leer sind
                  !            + ) Die Länge des Boardnamens wird
                  !            + ) auf 8 Zeichen abgeschnitten.
                  !            + )
                  !
                  +--GERAETE---+--YAESU
                  !            !
                  !            +--KENWOOD
                  !            !
                  !            +--ICOM
                  !
                  +--COMPUTER--+--ATARI
                  !            !
                  !            +--IBM
                  !            !
                  !            +--C64
                  !            !
                  !            +--AMIGA
                  !            !
                  !            +--APPLE
                  !
                  +--SOFTWARE--+--BAYCOM
                               !
                               +--GP
                               !
                               +--PAXON
```

- Bulletin-Namen, die nicht im (vom Sysop) angelegten Baum vorhanden sind, werden unter dem Verzeichnis TMP temporär angelegt. Sie gehorchen dabei allen Gepflogenheiten der anderen Verzeichnisse, mit dem Unterschied dass sie wieder gelöscht werden, sobald sie leer sind. Diese Automatik greift bei allen Verzeichnissen, die unterhalb von TMP angelegt werden.
- Einschränkung der baumförmigen Struktur: Jeder Bulletin-Name muss eindeutig sein. Bei der Ein- und Ausgabe der Mails verschwindet die baumförmige Struktur und es ist nur noch der Verzeichnis-Name entscheidend. D.h. man braucht beim Wechsel in ein Sub-Board nie den Namen des darüber liegenden Boards mit anzugeben, der Name des Sub-Boards reicht völlig.
- Die Dateinamen der Boxeinträge auf der Dateisystemebene setzen sich aus Datum und Uhrzeit zusammen, wann die Mail in der Box angekommen ist. Beispiel:

```
131N50A
JMTHMSN
      ^ laufende Nummer (0..9, A..Z)
     ^ (Bit0 der Minuten)*15 + Sekunden/4 (0..9, A..Z)
    ^ Minuten/2 (0..9, A..Z)
   ^ Stunden (0..9, A..Z)
  ^ Tag (0..9, A..Z)
 ^ Monat (Hex)
^ Jahr (1=1991, 2=1992, ..., A=2000, B=2001 usw.)
```

  Treffen mehrere Mails innerhalb von 4 Sekunden ein, so wird das letzte Zeichen im Namen hinaufgezählt. Gezählt wird im erweiterten Hexcode, also 0..9, A..Z.
- In jedem Verzeichnis wird eine Datei namens *list.bcm* aufgebaut. Diese Datei ist chronologisch sortiert. In ihr werden alle wesentlichen Daten der Mails in diesem Verzeichnis gehalten und können bei Bedarf ausgelesen werden. Diese Datei ist zwar in ASCII und lesbar, aber sehr

formsensibel und deshalb nicht ohne weiteres editierbar. Das Aussehen der Einträge ist folgendes:

```
4B42AA0<DL8MBT>RUNDSPRU~DB0LNA@DB0AAB$04B4OE3XSR0F#123%xxyyy|!!!!!!!!!!!!-Betr
|       |      |        |      |      |            |   | |   |
|       |      |        |      |      |            |   | |   16 Mail-Flags
|       |      |        |      |      |            |   | Bytes (komprimiert)
|       |      |        |      |      |            |   Zeilen (komprimiert)
|       |      |        |      |      |           Lifetime
|       |      |        |      |      Bulletin-ID (12 Zeichen)
|       |      |        |      Forward-Adresse (nur erster Teil der H-Adr)
|       |      |        Box, die die Nachricht geschickt hat (Nachbarbox)
|       |      Zieladresse, (Boardname oder Zielrufzeichen) 8 Zeichen
|       Absender (6 Zeichen, gültiges Afu-Call)
Datum des Eintreffens in der Box, Form JMTHMSN (lexikalisch sortierbar)
```

- Bei User-Mails (und nur bei diesen) wird hinter der Zieladresse ein R (durch ein Blank getrennt) angegeben, wenn die Nachricht vom Inhaber des Zielrufzeichens gelesen worden ist. Da bei Usermails das Ziel max. 6 Zeichen haben kann, ist dafür noch Platz.
- Das ">" vor dem Zielrufzeichen wird gegen ein Blank ersetzt, wenn die Nachricht als gelöscht markiert wird. In diesem Zustand kann die Nachricht wieder restauriert werden (mit dem Befehl UNERASE).
- Die ersten 7 Zeichen in jeder Zeile entsprechen zugleich dem Dateinamen der beschriebenen Mail.
- Die Zahl der Zeilen und die Zahl der Bytes werden in einem komprimierten Format angegeben: Zahlenwerte zur Basis 128, beginnend mit dem !-Zeichen (hex 21 im ASCII-Zeichensatz).
- Die Felder sind exakt einzuhalten. Sind die Feldelemente kürzer als die vorgesehene Länge, so sind Blanks einzufügen. Die Operatoren zu Beginn jedes Feldes werden stets eingefügt, aber nicht ausgewertet. Entscheidend ist allein die Lage der Felder. Zum Ende wird jeder Eintrag mit dem ersten Teil des Betreffs der Nachricht aufgefüllt, so dass sich eine Gesamtlänge von 128 Bytes ergibt. Der Betreff endet mit der Folge CR/LF.
  Achtung: Die Bearbeitung der Datei erfolgt ausschließlich als Binärdatei. Die Datei darf nicht mit CTRL-Z abgeschlossen sein, und es darf keine Konvertierung CR/LF zu CR oder LF erfolgen, um die fixe Strukturlänge von 128 Bytes keinesfalls zu verletzen. Aus diesem Grund ist das Bearbeiten mit einem Editor zu unterlassen!
- Die Kommandos LIST und DIR greifen auf das jeweilige *list.bcm* zu.
- Verzeichnis-Array: Um hinreichende Verarbeitungsgeschwindigkeit zu ermöglichen, wird der gesamte Verzeichnisbaum für die Info-Mails stets im RAM gehalten. Dabei werden die vollständigen Pfadnamen Abgespeichert und die Suche nach einzelnen Verzeichnissen wird zeitoptimiert implementiert. Es wird also bei allen Dateireferenzen nie der gesamte Pfad, sondern immer nur der Name selbst angegeben. Der Pfad kann dann diesem Array entnommen werden. Erforderlichenfalls kann eine Hash-Tabelle nach den Boardnamen hinzugefügt werden, was aber den Nachteil hat, dass nicht nach Namensfragmenten gesucht werden kann. Das Array hat folgende Form:

  ```
  HARDWARE
  GERAETE
  GERAETE/YAESU
  GERAETE/KENWOOD
  GERAETE/ICOM
  COMPUTER
  COMPUTER/IBM
  COMPUTER/ATARI
  [...]
  ```

- Die Sortierung erfolgt in der angegebenen Reihenfolge, sprich im Baum von oben nach unten. Dadurch ist sichergestellt, dass bei der Suche nach Fragmenten stets im eigentlichen Verzeichnisnamen eine Übereinstimmung auftreten kann.

24.3. Overview of all mailbox files and directories

Die Mailbox kann sich in einem beliebigen Verzeichnis auf der Festplatte befinden, üblicherweise wird das Verzeichnis *c:\bcm* (DOS/WinNT) bzw. */bcm* (Linux) verwendet. Die Verwendung eines anderen Verzeichnisses hat keine Auswirkung auf die Funktion der Box. Es ist lediglich unter Linux die Systemvariable "$BCMHOME=/bcm_path" einzustellen, wobei "/bcm_path" jener Pfad ist, in dem sich die Systemdateien der OpenBCM-Mailbox befinden. Es kann dann allerdings passieren, dass manche Runutils nicht mehr korrekt funktionieren.

afwd.bcm
Diese Datei wird vom Autorouter automatisch erzeugt und immer wieder verändert. Hier sind die dem Autorouter bekannten Ziele abgespeichert. Ein manuelles Editieren dieser Datei macht keinen Sinn!

asysop.bcm
In dieser Datei können die Rufzeichen festgelegt werden, die automatisch Sysop-Status in der Mailbox besitzen sobald sie connected sind. Das Format der Datei wird in Kapitel "5.6. Autosysop (asysop.bcm)" erklärt.

bcm
Die OpenBCM-Mailbox für Linux. Es können keine Parameter übergeben werden. Es empfiehlt sich außerdem, die OpenBCM-Mailbox mit dem Shell-Script *startbcm* zu starten.

bcm.exe
Die OpenBCM-Mailbox für DOS. Es können folgende Parameter übergeben werden:
- /?: Zeigt mögliche Optionen
- /C: Benutzt bei VGA 25 statt 50 Zeilen (spart 5 kB Speicher bei VGA)
- /M: Benutzt monochrom-Farbattribute
- /N: Es erfolgen keine Zugriffe auf die Videokarte
- /F: Zeigt die mögliche Anzahl offener Dateien; ist dieser Wert kleiner als 40, so ist die Zahl der "FILES" in der Datei *config.sys* zu erhöhen

Für einen ersten Testlauf wird im Unterverzeichnis *msg* mindestens noch die Datei *messages.gb* oder *messages.dl* benötigt.

bcm32.exe
Die OpenBCM-Mailbox für Windows. Nach dem Start öffnet sich ein Fenster mit Trace-Ausgaben. Die Mailbox selbst ist z.B. über Telnet (Standard: TCP-Port 4719) oder HTTP (Standard: TCP-Port 8080) erreichbar.

bct
ein primitives Terminalprogramm für Linux, mit dem man derzeit ausschließlich eine Verbindung zur Box selbst herstellen kann. Während das Programm läuft, darf ggf. die Fenstergröße nicht verändert werden, weil dies nicht abgefangen wird.

beachead.bcm
Falls diese Datei vorhanden ist, so wird deren Inhalt am Anfang jeder Mail-Bake ausgesendet.

beacon.bcm
Enthält die Digipeater-Pfade über welche die Mail-Bake abgestrahlt werden soll. Ist die Bake länger als ein Paket, wird sie in mehreren Paketen ausgesendet. Das Format dieser Datei ist bei DOS und Linux/Windows unterschiedlich!

bidh2.bcm

Diese Datei wird automatisch erzeugt und enthält die 32 Bit-Hash-Tabelle für den Zugriff auf *bids2.bcm* bzw. *bids3.bcm*.

bids2.bcm bzw. ***bids3.bcm*** bei der Linux-Version

Diese Datei wird automatisch erzeugt und enthält die letzten empfangenen und erzeugten Bulletin-IDs. Die Größe dieser Datei darf beliebig anwachsen, bei jedem Reorg wird sie auf MAXBIDS begrenzt.

boardinf.bcm

Diese Datei kann Beschreibungen zu den einzelnen Rubriken enthalten, die bei DIR oder LIST ausgegeben wird. Diese Datei braucht aber nicht vorhanden oder vollständig sein um einen reibungslosen Boxbetrieb zu gewährleisten.

bulletin.bcm

Enthält alle verfügbaren Bulletin-Boards. Dabei beginnen Hauptboards in der ersten Spalte, Subboards sind um ein Leerzeichen eingerückt. Nach dem Boardnamen steht durch mindestens ein Blank getrennt die Lifetime in Tagen. Die Datei wird beim Start der Boxsoftware gelesen und bei einer Änderung erneut auf die Platte geschrieben. Während die Box läuft bringt eine Änderung der Datei nichts, weil die Datei regelmäßig wieder aktualisiert wird.
Achtung: Diese Datei nicht unbekümmert editieren, sondern Änderungen nur über die Befehle LIFETIME, MKBOARD, RMBOARD, MVBOARD vornehmen, da sonst Inkonsistenzen entstehen können.
Wenn die OpenBCM-Mailbox nicht läuft, so ist es ohne weiteres möglich Boards hinzufügen oder eine Lifetime direkt in der Datei zu ändern. Beim Löschen oder Verschieben von Boards muss allerdings sichergestellt werden, dass das Board leer ist, sonst werden dessen Mails unsichtbar und fallen auch nicht mehr der Lifetime zum Opfer.
Ist die Box leer (vor der ersten Inbetriebnahme), kann die Datei hingegen gefahrlos und nach Herzenslust editiert werden.
In Störungsfällen kann *bulletin.bcm* beschädigt werden. Dann ist es entweder erforderlich, die Datei zu editieren oder ein Backup einzuspielen. Damit dies wirkt, ist folgende Vorgehensweise notwendig:

- disable (no further user logins are possible)
- disconnect all user tasks with KILL command or do a SHUTDOWN, to make sure that no user is in the mailbox
- wt bulletin.bcm
- upload new file
- SHUTDOWN (shutdown your mailbox because the file is only read in when the mailbox is starting)

If the file *bulletin.bcm* does not exist when the mailbox is starting, a new file is created with the found directory structure.
The lifetime settings and empty boards are lost in this case and for security reasons the default lifetime "999" is used.

check.bcm

Diese Datei wird automatisch erzeugt. Hier wird die (sehr große) Liste für den CHECK-Befehl gespeichert. An die Liste werden eintreffende Bulletin-Nachrichten angehängt. Kommt die Liste durcheinander, so kann sie einfach gelöscht werden. Sie wird dann ggf. automatisch neu erzeugt. Dieser Vorgang dauert jedoch eine Weile, das Löschen sollte daher nicht unbekümmert erfolgen. Diese Liste sollte nicht editiert werden!

checknum.bcm

Diese Datei wird automatisch erzeugt und enthält die Board-Nummern für die Checkliste.

convat.bcm

Dient zur Konvertierung von Verteilern. Lokal eingegebene oder im User-S&F empfangene Verteiler werden konvertiert. Beim S&F mit einem offiziellen Forward-Partner wird der Verteiler bei der Auswertung der Zielboxen konvertiert, d.h. alte Verteiler brauchen nicht mehr in *fwd.bcm* stehen. Ein Beispiel ist im Kapitel "5.12. Automatic director conversion (convat.bcm)" angegeben.

Mit dem Befehl CONVAT kann man sich diese Datei ausgeben lassen.

convert.bcm

In dieser Datei können Rubriknamen von ankommenden Mails umbenannt werden. Dadurch lässt sich die Flut der beim Forwarding entstehenden Rubriken eindämmen und es lassen sich Lifetime-Konvertierungen durchführen. Bei der Weitergabe der Nachrichten an die entsprechenden Partner-Forwardboxen werden jedoch wieder die ursprünglichen Rubriknamen verwendet. Der Aufbau und die Funktionsweise der Datei *convert.bcm* ist im Kapitel "5.11. Automatic board conversion (convert.bcm)" näher beschrieben.

Wird der Inhalt der Datei extern verändert, so kann sie mit NEW neu eingelesen werden; bequemer ist die Verwendung der CONVEDIT Funktion.

convlife.bcm und ***convname.bcm***

Diese Dateien wurden in alten Versionen zur Rubrikkonvertierung eingesetzt. Sie werden in den aktuellen Versionen nicht mehr verwendet.

crontab.bcm

Enthält die Zeitpunkte, zu denen verschiedene Boxfunktionen aufgerufen werden. Diese Datei sollte während die Box läuft nur mit boxinternen Befehlen (EDIT, RTEXT, WTEXT) bearbeitet werden.

cvusers4.exe

DOS-Programm zum Umwandeln von alten Userdatenbanken (*users.bcm* bzw. *users3.bcm*) in das aktuelle Format *users4.bcm*. Die Umwandlung wird normalerweise direkt von der Mailboxsoftware durchgeführt, sofern man denn bei dem gleichen Betriebssystem geblieben ist. Somit wird im Regelfall dieses Tool eigentlich nicht benötigt wird.

Ein Spezialfall ist jedoch der Umstieg von z.B. einer alten DOS-Version, die noch *users.bcm* benutzt, auf die aktuellste Linux-Version, die *users4.bcm* benutzt. Hier kann man dann mit diesem Tool die Datenbank manuell umstellen. Zur Konvertierung ist einfach das Programm im DOS oder Windows-OpenBCM-Verzeichnis zu starten.

fwd.bcm

Hier stehen alle eingestellten Forwardwege. Diese Datei ist komplett vom Sysop anzulegen und wird auch von der Mailbox nicht verändert. Es kann im Betrieb mit dem Befehl NEW neu eingelesen werden. Im Kapitel "9. Complete forward configuration" wird der Aufbau und die Funktionsweise dieser Datei ausführlich behandelt.

hadr4.bcm

Diese Datei wird automatisch erzeugt und enthält gespeicherte Mailboxadressen. Jeder ankommenden Nachricht werden Daten über durchlaufene Boxen abgenommen, die hier gespeichert werden. Jede Box belegt 1024 Bytes. Diese Datei wird von der Box automatisch erzeugt und kann nicht kann nicht aus vorhandenen Datenbeständen restauriert werden. Sie sollte deshalb regelmäßig gesichert werden!

hadrhash.bcm

Die Datei wird automatisch erzeugt und enthält die 16-Bit-Hash-Tabelle für den Zugriff auf *hadr4.bcm*.

init.bcm

Enthält die Grundeinstellungen der Box (muss also editiert werden!).

Während die Box läuft, bringt eine Änderung der Datei gar nichts, weil diese Datei von der Mailbox selbst wieder überschrieben wird. Änderungen müssen bei laufender Box direkt in der Box (als Kommandos) vorgenommen werden, diese werden dann automatisch in *init.bcm* zurück geschrieben.
Siehe dazu Kapitel "24.4. Initialization *file* *init.bcm*".

init.l2
In der Linux- und in der Windows-Version der OpenBCM-Mailbox ist auch die Funktion eines Layer2 enthalten. Dieser wird über diese Datei konfiguriert.
Siehe dazu Kapitel "24.5. Initialization *file* *init.l2*".

issue.bcm
Falls diese Datei vorhanden ist, so wird deren Inhalt vor dem "Login:"-Prompt beim Login per Modem (bei der DOS-Version) bzw. über Telnet (bei der Linux- und Windows-Version) ausgegeben.

macro.bcm
Die Verbindung zwischen Macro-Ereignissen und Skriptnamen wird in dieser Datei festgelegt.
Siehe hierzu auch das Kapitel "29. OpenBCM script language".

netpath.bcm
Enthält Connectpfade für das Net-CMD-Interface unter Linux, siehe hierzu auch das Kapitel "19.8. Net-CMD interface ".

passwd.bcm
Enthält die Passwörter für die Sysop-Identifikation mittels BayCom-, MD2- und MD5-Passwort-Verfahren. Für jedes Verfahren ist ein eigenes Passwort anzugeben. Die Passwörter stehen jeweils in einer eigenen Zeile in der angegebenen Reihenfolge.

popstate.bcm
Datei enthält Datenbank zu POP3/SMTP-Zugriffen.

pw
Ein Programm unter Linux, um über Funk einen "root"-Login zu ermöglichen, ohne dass das su-Passwort über Funk gesendet werden muss. Das dabei verwendete Verfahren entspricht exakt dem PW-Verfahren der OpenBCM-Mailbox. Das Programm lebt von der Datei */etc/passwd.pw*, in dem in der ersten Zeile ein 80-stelliger String (wie in *passwd.bcm*) steht, in der zweiten Zeile steht ein Prompt, der vor die 5 ausgegebenen Ziffern geschrieben wird. Das Programm braucht root-Rechte, um zu funktionieren. Das Programm wird nicht benötigt, wenn OpenBCM direkt als root-Prozess ausgeführt wird.

<Rufzeichen>.pwd, z.B. ***db0zdf.pwd***
Diese Datei enthält ggf. ein DieBox-kompatibles Passwort für Forwarding (auch User-S&F). Besser sollte das BayCom-, oder noch besser das MD2/MD5-Passwort-Verfahren verwendet werden.

reject.bcm
Diese Datei enthält die REJECT/HOLD Definitionen, die im Kapitel "5.10. Reject and Hold (reject.bcm)" ausführlich erläutert werden.

rhosts.bcm
In dieser Datei können beliebig viele Hostnamen und IP-Adressen angegeben werden (jeweils in einer eigenen Zeile). Wenn das Telnet-Interface der Mailbox von einem Host connected wird, der in dieser Datei angegeben ist, dann kann sich der Benutzer ohne Passwort einloggen und erhält Sysop-Status!

Achtung: Während der DNS-Abfrage steht die Mailbox, hat also der Nameserver eine wackelige Anbindung, so ist es sinnvoll die Anzahl der Einträge in dieser Datei gering zu halten.

rundat.bcm
Wird vor dem Aufruf eines RUN-Utilities erzeugt. Diese enthält Informationen für das aufgerufene Run-Utility.

runutil.bcm
Enthält Informationen über externe Programme, die von Usern aufgerufen werden können, es handelt sich dabei um so genannte Run-Utilities.

speech.bcm
Diese Datei beinhaltet die Sprachdatei-Zuordnungen und wird im Kapitel "5.8. Multilanguage configuration (speech.bcm)" erläutert.

startbcm
Shell-Skript unter Linux zum Starten der Mailbox. Hier werden auch alle notwendigen Umgebungsvariablen vorbereitet, die im Betrieb der Box sichtbar sein sollten. Der Aufruf sollte sinnvollerweise aus */etc/inittab* erfolgen.

userh2.bcm
Diese Datei wird automatisch erzeugt und enthält die 16 Bit-Hash-Tabelle für den Zugriff auf *users4.bcm*.

userpw.bcm
Diese Datei wird durch den Befehl PWGEN automatisch erzeugt und enthält mögliche User-Passwörter, die der Sysop mit SETPW einem User zuordnen kann.

users.bcm und ***users3.bcm***
Diese Dateien wurden in alten Versionen eingesetzt. Sie werden in den aktuellen Versionen durch *users4.bcm* ersetzt und werden nicht mehr verwendet.

users4.bcm
Diese Datei wird von der Box automatisch generiert und enthält die Daten der bekannten Benutzer, die sie im Laufe der Zeit lernt. Diese Datei kann nicht aus anderen Datenbeständen abgeleitet werden, deshalb sollte von dieser Datei regelmäßig ein Backup erstellt werden!

Verzeichnis ***fwd/***
In diesem Verzeichnis werden für jeden Forward-Partner Dateien angelegt, in denen festgehalten wird, welche Mails zu diesem Partner weitergeleitet werden müssen. Sobald die Dateien nicht mehr benötigt werden, werden sie automatisch gelöscht.

Verzeichnis ***http/***
Hier werden Dateien abgelegt, die über den HTTP-Server abgerufen werden können. Außerdem kann hier *back.jpg*, *logo.gif* bzw. *qsl.wav* ablegen, die das HTTP-Erscheinungsbild verändern können.

http/back.jpg
Diese Datei wird von der Box als Hintergrundbild bei HTTP-Zugriff genutzt, falls vorhanden

http/qsl.wav
Diese Datei wird von der Box als Quittierungston bei jeder HTTP-Aktion ausgegeben, falls vorhanden.

http/logo.gif

Diese Datei wird von der Box als Logo-Bild bei HTTP-Zugriff oben Links angezeigt, falls vorhanden.

Verzeichnis ***msg/***
In diesem Verzeichnis werden alle System-Sprach-Dateien gespeichert. Sie sind ggf. vom Sysop zu editieren und den Gegebenheiten anzupassen. Die Extension <Sprachkenner> bezeichnet den Landeskenner der eingestellten Sprache. Für jede Sprache ist mindestens die Datei *msg/messages.<Sprachkenner>* notwendig.

msg/messages.<Sprachkenner>, z.B. ***msg/messages.gb***
Enthält alle Meldungen der Box in der jeweiligen Sprache.

msg/help.<Sprachkenner>, z.B. ***msg/help.gb***
Enthält die Onlinehilfe. Diese Datei muss existieren, damit HELP Funktioniert.

msg/aktuell.<Sprachkenner>, z.B. ***msg/aktuell.gb***
Enthält den Text, den jeder genau einmal beim Login gezeigt bekommt. D.h. wenn der Text neuer ist als der letzte Login, wird er angezeigt. Mit dem Befehl AKTUELL kann der Text jederzeit angesehen werden.

msg/bnguest.<Sprachkenner>, z.B. ***msg/bnguest.gb***
Enthält den Text, der bei dem Login als Gast im CB-BCMNET Loginkonzept (CB-Funk) ausgegeben wird.

msg/bnpwonly.<Sprachkenner>, z.B. ***msg/bnpwonly.gb***
Enthält den Text, der bei einem Login ohne Passwort im CB-BCMNET Loginkonzept (CB-Funk) ausgegeben wird.

msg/cein.<Sprachkenner>, z.B. ***msg/cein.gb***
Enthält den Text, der beim Login eines Users ausgegeben wird, wenn für diesen User die ASKLOGIN-Abfrage ausgeführt wird (bei "asklogin 1" in *init.bcm*).

msg/cguest.<Sprachkenner>, z.B. ***msg/cguest.gb***
Enthält den Text, der bei dem Login unter GUESTCALL ausgegeben wird.

msg/cnew.<Sprachkenner>, z.B. ***msg/cnew.gb***
Enthält den Text, der beim Login eines Users ausgegeben wird, wenn für diesen User Helplevel 2 eingestellt ist.

msg/cstat4.<Sprachkenner>, z.B. ***msg/cstat4.gb***
Enthält den Text, der bei User-Status 4 ausgegeben wird.

msg/ctext.<Sprachkenner>, z.B. ***msg/ctext.gb***
Enthält den Text, der bei jedem Login eines Users ausgegeben wird.

msg/fshelp.<Sprachkenner>, z.B. ***msg/fshelp.gb***
Enthält den Text, der bei HELP im Filesurf ausgegeben wird.

msg/fsintro.<Sprachkenner>, z.B. ***msg/fsintro.gb***
Enthält den Text, der beim Aufruf des Filesurf-Modus ausgegeben wird.

msg/fsshelp.<Sprachkenner>, z.B. ***msg/fsshelp.gb***
Enthält den Text, der bei HELP im Filesurf ausgegeben wird, falls das eingeloggte Rufzeichen Sysopstatus (erweiterte Befehle) besitzt.

msg/ftpintro.<Sprachkenner>, z.B. ***msg/ftpintro.gb***
Enthält den Text, der bei FTP-Connect als erstes ausgegeben wird.

msg/info.<Sprachkenner>, z.B. ***msg/info.gb***

Dieser Text wird beim Befehl INFO ausgegeben.

msg/nousr.<Sprachkenner>, z.B. ***msg/nousr.gb***
Dieser Text wird bei einem Loginversuch ausgegeben, wenn in der Mailbox der Userzugriff ausgeschaltet wurde.

msg/pwonly.<Sprachkenner>, z.B. ***msg/pwonly.gb***
Dieser Text wird bei einem Login ohne Passwort und einem Wert größer 2 von USERPW in *init.bcm* ausgegeben.

msg/qtext.<Sprachkenner>, z.B. ***msg/qtext.gb***
Enthält den Text, der bei jedem QUIT ausgegeben wird. Dieser Text ist normalerweise sowohl unsinnig als auch unwichtig und sollte deshalb nur bei einem dringenden Bedürfnis eingesetzt werden.

msg/helpidx.<Sprachkenner>, z.B. ***msg/helpidx.gb***
Diese Datei wird automatisch generiert und enthält den Index für die Help-Datei in der jeweiligen Sprache.

Verzeichnis ***user/***
Hier werden alle User-Mails abgelegt. Jeder User erhält ein eigenes Verzeichnis, wenn für ihn Nachrichten vorliegen. Der Verzeichnis-Name kann in *init.bcm* geändert werden.

Verzeichnis ***info/***
Hier werden alle Bulletins abgelegt. Jedes Board erhält ein eigenes Verzeichnis, wenn Nachrichten vorliegen. Der Verzeichnis-Name kann in *init.bcm* geändert werden.

list.bcm
Diese Datei wird automatisch für jedes Board (also in den Verzeichnissen *info/* und *user/*) der Mailbox erzeugt. Sie enthält für jede Mail des Boards Angaben, wie Absender-Call, Titel, etc.

Verzeichnis ***log/***
In diesem Verzeichnis werden tagesweise die Logdateien abgelegt.

log/log<Datum>.bcm, z.B. ***log/log21030.bcm***
Diese Datei enthält das Logbuch für Box-Benutzung und wird von der Mailbox selber angelegt. Pro Tag wird eine Datei erzeugt, die das Datum in der Form JMMTT im Dateinamen hat. Vom Jahr wird jeweils nur die letzte Ziffer verwendet. Beispiel: *log21030.bcm* ist das Logbuch vom 30. Oktober 2002. Der User-Befehl LOG greift auf diese Dateien zu. Die Dateien können problemlos gelöscht werden, wenn sie mit der Zeit zu viel Platz fressen.

Verzeichnis ***macro/***
In diesem Verzeichnis werden die Skripte der Makrosprache abgelegt. Siehe hierzu auch das Kapitel "29. OpenBCM script language".

Verzeichnis ***temp/***
In diesem Verzeichnis befinden sich temporär von der Mailbox erzeugte Dateien. Spätestens nach dem Beenden der Mailboxsoftware kann man das Verzeichnis gefahrlos aufräumen.

Verzeichnis ***trace/***
In diesem Verzeichnis befinden sich von der Mailbox erzeugte Dateien, die (Fehler-)Meldungen, Forward-Protokolle und Mitschnitte von User-/Sysop-Eingaben enthalten.

trace/cmdlog.bcm
Hier werden alle vom Sysop eingegebenen Kommandos mit Datum und Uhrzeit gespeichert. Mit dem Sysop-Befehl CLOG kann man sich die

letzten zwei kByte dieser Datei ausgeben lassen.
Beispiel:

```
30.09.02 14:21z DH8YMB: par box
30.09.02 14:21z DH8YMB: d b
30.09.02 14:21z DH8YMB: mkb /tmp
30.09.02 14:22z DH8YMB: mkb /baybox
30.09.02 14:22z DH8YMB: reorg
30.09.02 14:22z DH8YMB: purge
30.09.02 14:22z DH8YMB: v
```

trace/eraselog.bcm

Hier werden alle gelöschten Mails protokolliert. Mit dem Sysop-Befehl ERLOG kann man sich die letzten zwei kByte dieser Datei ausgeben lassen.
Beispiel:

```
03.01.03 13:03:08z DBO251: #F DBO274>WP       31DDBO27400Q WP Update
03.01.03 13:35:11z DBX387: #F DBO274>W        31DDBO27400S WP Update
03.01.03 13:58:56z GPN80T: #F DBO274>W        31DDBO27400U WP Update
03.01.03 13:59:36z KS1BBS: #F DGT274>AT5HPK   31DDBO27400T beta20...
03.01.03 14:05:48z DBQ343: #F DBO274>WP       31DDBO27400V WP Update
03.01.03 14:51:15z DBQ946: #F DBO274>W        31DDBO27400W WP Update
03.01.03 15:05:49z DBO812: #F DBO274>W        31DDBO27400Y WP Update
03.01.03 15:06:01z DBX570: #F DBO274>W        31DDBO27400X WP Update
03.01.03 15:35:16z DBO251: #F DBO274>WP       31DDBO27400Z WP Update
```

trace/m_filter.bcm

Hier werden alle M_FILTER Aufrufe mitprotokolliert.

trace/oldumail.bcm

Hier werden alle Usermails mitprotokolliert, bei denen die Lifetime abgelaufen ist, bevor sie vom Empfänger gelesen wurden und bei denen der Absender der Mail mit einer Benachrichtigungsmail über den UNREAD Status der Mail informiert wurde.
Beispiel:

```
27.08.02 03:05:00z D81EUR > SAR0CA @SAR0CA Re: test
26.11.02 03:05:00z DGT274 > DGT882 @DBO274 test
```

trace/pwlog.bcm

Hierbei handelt es sich um die Passwort-Logdatei. Darin werden alle Passwort-Eingaben mitprotokolliert, egal ob erfolgreich oder erfolglos. Mit dem Befehl PWLOG kann man sich die letzten zwei kByte dieser Datei ausgeben lassen.
Beispiel:

```
30.09.02 12:18z OE3DZW via TCP/IP    tty login failure
30.09.02 12:19z DL8MBT via TCP/IP    tty login failure
```

trace/rejlog.bcm

In dieser Datei werden alle abgelehnten Mails protokolliert. Mit RLOG kann sich die letzten 2 kByte ausgeben lassen.
Beispiel:

```
16.12.02 22:22:25z B DGT274>TEST    @~ $GCCDBO274016 only message to SYSOP without PW
16.12.02 22:24:53z B DGT274>TEST    @~ $GCCDBO274017 only message to SYSOP without PW
16.12.02 22:28:52z B DGT111>TEST    @~ $GCCDBO27401A only message to SYSOP without PW
02.01.03 16:02:30z B PD2RLD>WHY?    @EU~NL3DHG$3352-PD2RLD  board not wanted
```

trace/sfhold.bcm

Hier werden Mails eingetragen, die aufgrund eines Forward-Fehlers nicht weitergeleitet wurden. Mit dem Befehl SFHOLD kann man sich die letzten zwei kByte dieser Datei ausgeben lassen.
Je nach Ursache wird ein Flag gesetzt:

- #R: Reject - Mail wurde vom Forward-Partner mit REJ abgelehnt
- #S: Size - Mail zu Gross (bei gesetzter Option -B in *fwd.bcm*)
- #B: Binary - Forward-Partner unterstützt AutoBIN nicht
- #E: Error - Nachricht ist fehlerhaft (z.B. fehlende BID bei Bulletins)
- #K: kein binär Bulletin Forward (Option -K in *fwd.bcm*)
- #L: Loop - Schleife (nur bei P)
- #M: kein 7plus Bulletin Forward (Option -M in *fwd.bcm*)

- #N: No - eine User-Mail wurde vom Partner abgelehnt, obwohl noch nie angeboten (von Partner welcher nicht REJ unterstützt)
- #H: Hold - eine Mail wird nur aufgrund eines Forwardfehlers auf HOLD gesetzt.

Beispiel:

```
02.01.03 08:54:08z #R DBX387:IT5TNT > SOFT   @ EU $65342_IT5TNT rejected
02.01.03 16:02:38z #R DBX387:RF76WI > TELNET @ EU $5921_RF76WI rejected
02.01.03 16:02:38z #R DBO812:RF76WI > TELNET @ EU $5921_RF76WI rejected
02.01.03 16:02:49z #R NL3DHG:DAA731 > DIVERS @ EU $16425_MG1BOX rejected
02.01.03 16:03:42z #R NL3DHG:DBX353 > Z      @ THEBOX $11DDBX353001 rejected
```

trace/swap.bcm

In dieser Datei werden all die Mails protokolliert, die offensichtlich von einer ANDEREN Mailbox ge"swap"t wurden. Diese Nachrichten bekommen automatisch eine Lifetime von 2 Tagen, während denen der Sysop Nachforschungen anstellen kann und bei Bedarf die Lifetime erhöhen kann. Mit dem Befehl SWAPLOG kann man sich die letzten zwei kByte dieser Datei ausgeben lassen.

Beispiel:

```
23.12.02 10:15:28z AT5HPK>BAYCOM  @DEU~DBQ946$NCCAT5XBR00Z to: BAYBOX
27.12.02 20:51:44z MW18PR>INET    @DEU~GPN80T$NCCLB7BOX00E to: WWW
27.12.02 20:51:44z MW18PR>INET    @DEU~GPN80T$NCCLB7BOX00E @: SAX
27.12.02 20:51:46z MW18PR>PR      @DEU~GPN80T$NCCLB7BOX00D to: PACKET
27.12.02 20:51:46z MW18PR>PR      @DEU~GPN80T$NCCLB7BOX00D @: SAX
01.01.03 02:49:01z AT6HSB>XNET    @DEU~DBO812$11DDBX507005 @: ALLE
01.01.03 02:51:48z AT6HSB>XNET    @DEU~DBO812$11DDBX507006 @: ALLE
```

trace/syslog.bcm

Ablage aller System-Fehlermeldungen. Wenn die Box einen internen Fehler entdeckt, wird er hier vermerkt. Mit dem Befehl SLOG kann man sich die letzten zwei kByte dieser Datei ausgeben lassen.

Beispiel:

```
30.09.02 12:17:52z SYSTEM: #L mbmain: start V1.03 (mem 8876kB, disk 205MB)
30.09.02 12:17:59z SYSTEM: #L cron: crontab.bcm generated
30.09.02 12:18:50z SYSTEM: #L logind: stop (SIGINT)
```

trace/syslog_r.bcm

Bei TRACELEVEL 0 werden hier alle Report-Meldungen abgelegt. Mit dem Sysop-Befehl SLR kann man sich die letzten zwei kByte dieser Datei ausgeben lassen.

Beispiel:

```
30.09.02 12:17:52z SYSTEM: #R init_tnc: 200 ports
30.09.02 12:17:52z SYSTEM: #R readtree: 0 bulletins
30.09.02 12:17:53z SYSTEM: #R socketd: bcm not found in /etc/services
30.09.02 12:17:53z SYSTEM: #R socketd: initialised to port 4719
```

trace/unknown.bcm

Hier werden unbekannte Forwardadressen vermerkt mit Hinweis auf die Mails, die es betrifft. Mit dem Befehl UNKNOWN kann man sich die letzten zwei kByte dieser Datei ausgeben lassen.

In dieser Datei werden folgende Informationen vermerkt:

- Datum / Uhrzeit
- Login-Rufzeichen (also Boxcall oder Absender)
- Absenderrufzeichen
- Zielrufzeichen oder Boardname
- Forwardadresse (unbekannte Adresse, um die es geht)
- Erster Teil vom Betreff der Nachricht.

trace/t_<Rufzeichen>.bcm, z.B. ***trace/t_hb9w.bcm***

Wenn in *fwd.bcm* für einen Forward-Partner die Option "-T" eingetragen ist, und der Parameter FWDTRACE in der Datei *init.bcm* auf den Wert 2 gesetzt ist, wird das Forwarding mit diesem Partner in dieser Datei genau mitprotokolliert.

Ist in *init.bcm* der Parameter FWDTRACE auf den Wert 1 eingestellt, dann werden alle Forward-Verbindungen mitprotokolliert, unabhängig von der Einstellung in *fwd.bcm*!

Durch Setzten des Parameters FWDTRACE auf 0 wird das Mitprotokollieren von Forward-Verbindungen komplett abgeschaltet.

Die Dateien *users4.bcm* und *hadr4.bcm* werden zwar von der Mailbox erzeugt, können aber nicht aus anderen Daten restauriert werden, falls sie verloren gehen. Sie sollten deshalb regelmäßig gesichert werden!

Die Dateien *bids.bcm*, *bidhash.bcm*, *haddress.bcm*, *badnames.bcm*, *convname.bcm*, *convlife.bcm*, *users.bcm*, *users3.bcm*, *hadr2.bcm* und *hadr3.bcm* sind Überbleibsel von alten Versionen der OpenBCM/BCM-Mailbox. Sie werden nicht mehr verwendet und können deshalb gefahrlos gelöscht werden.

24.4. Initialization file *init.bcm*

Diese Datei enthält die Parameter der Mailbox, die unmittelbar nach dem Start gesetzt werden. Sie können vom Sysop während des Betriebs wie ganz normale Box-Befehle eingegeben werden. Jede Änderung wird sofort in die Datei geschrieben. Eingelesen wird diese jedoch nur beim Start, eine Änderung direkt in der Datei während des Betriebs bringt also nichts.

Alle Parameter der Datei *init.bcm* werden nachfolgend genauer beschrieben:

Abschnitt "Mailbox":

BOXADDRESS	Hierarchische Adresse der Mailbox, z.B.: DB0IRS.#BAY.DEU.EU
BOXHEADER	legt den Text der bei einer "Sent:"-Zeile im Kopf einer Mail der eigenen Box erscheint, z.B.: OpenBCM Oelde JO41BU
CALLFORMAT	Legt Amateurfunk/CB-Version incl. Rufzeichen-Check fest: 0 = Amateurfunkversion, nur Amateurfunkrufzeichen sind erlaubt 1 = CB-Version, nur CB-Funkrufzeichen sind erlaubt 2 = CB-Version, CB- und Amateurfunkrufzeichen sind erlaubt
FSPATH	Pfade, die für User im Filesurf verfügbar sind
FSBMAIL	Legt die Verfügbarkeit des BMAIL Befehl im Filesurf fest: 0 = Befehl ist nicht verfügbar 1 = Befehl ist verfügbar
GUESTCALL	Legt ein Gastrufzeichen fest, z.B.: GAST
INFOPATH	Pfad, in dem die Bulletin-Mails abgespeichert werden
LOGINCALLFORMAT	Legt Rufzeichen-Check bei Option _LCF in *config.h* fest: 0 = Amateurfunkrufzeichen sind erlaubt 1 = CB-Funkrufzeichen sind erlaubt 2 = CB- und Amateurfunkrufzeichen sind erlaubt
MAXBIDS	Anzahl der BIDs/MIDs, die gespeichert werden, z.B. 5000000
REMERASE	Erlaubt das Löschen von Mails via Remote Erase
SAVEBROKEN	Konfiguriert den Empfang von 7+ Mails
SYSOPCALL	Call des Sysops - kann sich an der Konsole mit "." einloggen
USERPATH	Pfad, in dem die User-Mails abgespeichert werden

Abschnitt "Packet Interface":

AX25K_IF	HW-Adresse eines AX.25-Netzwerk-Interfaces (nur Linux)
FWDSSID	SSID, unter dem ausgehende S&F-Connects gemacht werden
MYCALL	AX.25-Calls der Mailbox, z.B. DB0ABC-8 DB0ABC-7

Abschnitt "TCP/IP Interface":

HTTP_PORT	TCP-Port für HTTP, z.B. 8080
FTP_PORT	TCP-Port für FTP, z.B. 8021
NNTP_PORT	TCP-Port für NNTP, z.B. 8119
POP3_PORT	TCP-Port für POP3, z.B. 8110
SERV_PORT	TCP-Port für das Service Interface, z.B. 8139
SMTP_PORT	TCP-Port für SMTP, z.B. 8025
TELNET_PORT	TCP-Port für TELNET, z.B. 4719
INTERNETMAILGATE	Steuert die mögliche Funktion des SMTP-Gateways für CB-Funk
NOPOPSMTP	Legt SMTP-Zugriff fest:

	0 = SMTP ist nur nach vorherigem POP3 möglich
	1 = SMTP ist auch ohne vorheriges POP3 möglich
HTTPACCOUNT	Bei "1" ist bei HTTP-Zugriff "create account" aktiv; besser "0"
HTTPGUESTFIRST	Bei "1" ist bei HTTP-Zugriff zunächst GUESTCALL eingeloggt
HTTPROBOTS	Bei "1" dürfen Webrobots von Suchmaschinen die Mailbox indexieren, bei "0" ist es diesen untersagt
HTTPTTYPW	Bei "1" wird immer das TTYPW abgefragt, auch für das AMPRNET
UNSECURETTYPW	Konfiguriert den Zugriff auf ALTER TTYPW

Abschnitt "Timers":

AUTOFWDTIME	Legt die Zeit fest, die vom Autorouter ausgewertet wird
FWDTIMEOUT	Timeout bei S&F-Verbindungen
HOLDTIME	Legt die Haltezeit für Mails mit HOLD-Aktion (*reject.bcm*) fest
INFOLIFE	Lifetime von Mails in TMP-Board
OLDESTBEACON	Legt fest, wie lange Usermails in der Mailbake angezeigt werden
OLDESTFWD	Maximales Alter von Mails, bis zu dem sie geforwardet werden
USERLIFE	Lifetime der User-Mails
USRTIMEOUT	Legt ein User-Timeout fest

Abschnitt "Logging":

ERASELOG	Legt fest, in wie weit Löschungen mitprotokolliert werden
FWDTRACE	Legt fest, in wie weit der S&F mitprotokolliert wird
TRACELEVEL	Legt fest, welche Status-Meldungen mitprotokolliert werden
TCPIPTRACE	Legt fest, in wie weit TCPIP-Ausgaben mitprotokolliert werden

Abschnitt "System":

DOSINPUT	Definiert die Ausführung eines DOS-Kommandos (nur unter DOS)
DISABLE	Bei "1" werden keine neuen Logins mehr zugelassen
SFONLY	Aktiviert/Deaktiviert reine S&F-Mailbox
TIMESLOT	interne Variable zur Tasksteuerung
TTYMODE	Parameter für die Ansteuerung der seriellen Schnittstelle
USVSENSE	Shutdown der Box durch USV-Warnung (nur unter DOS)
WATCHDOG	Konfiguriert den Watchdog (nur unter DOS)
ADDLINUXSYSTEMUSER	
	Support für Linux-User-Password aktivieren/deaktivieren

Abschnitt "User constraints":

ALTBOARDINFO	Board-Info aus *boardinf.bcm* mit/ohne Return ausgeben
ASKLOGIN	Abfrage von Name, QTH, ZIP-Code und MYBBS bei unbekannten Usern
CREATEBOARD	Werden noch nicht vorhandene Boards automatisch angelegt?
LTQUERY	Bei "1" werden Mails nur mit Lifetimeangabe akzeptiert
M_FILTER	Legt ein Mailfilter-Programm fest
MAILBEACON	Bei "1" wird die Mailbake immer gesendet; besser auf "0" setzen
MAXLOGINS	Begrenzt die Anzahl der Logins mit gleichem Call in die Mailbox
MAXPACLEN	Legt den maximalen User-Paclen-Wert fest (für ALTER PACLEN)
MINPACLEN	Legt den minimalen User-Paclen-Wert fest (für ALTER PACLEN)
NOPURGE	Legt fest, in wie weit Mails beim PURGE gelöscht werden
NOUNKNOWNROUTE	Bei "1" werden Mails mit unbekannter Route abgelehnt
PACLEN	Stellt die globale Paclen ein, bei "0" Paclen-Funktion deaktiv
READLOCK	Gibt ALTER READLOCK für User frei
SMOOTHHEADER	Bei "1" verkürzter Mailheader; besser auf "0" setzen
TIMEOUTWARNING	Bei "1" Timeout-Warnung an den User
USERPW	Legt fest, ob sich User ein Passwort setzen dürfen
USERQUOTA	Anzahl der kBytes, die ein User täglich schreiben/lesen darf

Abschnitt "Server":

DEFSWAPLF	Legt Lifetime von als ge"swap"ped erkannten Mails fest
MAILLISTSERV	Konfiguriert/Aktiviert den Mailing-Listserver
MAILLISTSENDER	Konfiguriert das Absendercall einer Mailing-Listserver-Mail
OLDUMAIL	Bei "1" Info an den Sender einer Usermail, falls Mail ungelesen
POCSAGLISTSERV	Konfiguriert/Aktiviert den Pocsag-Server
TELLMODE	Konfiguriert/Aktiviert den Tell-Server

Abschnitt "User defaults":

```
DEFCHECK         Default-Einstellungen für ALTER CHECK
DEFCMD           Default-Einstellungen für ALTER COMMAND
DEFFBBCHECKMODE  Default-Einstellungen für ALTER FBBCHECKMODE
DEFHELP          Default-Einstellungen für ALTER HELPLEVEL
DEFHOLD          Default-Einstellungen für ALTER FHOLD
DEFHTTPSURFACE   Default-Einstellungen für ALTER HTTPSURFACE
DEFIDIR          Default-Einstellungen für ALTER IDIR
DEFILIST         Default-Einstellungen für ALTER ILIST
DEFIREAD         Default-Einstellungen für ALTER IREAD
DEFLF            Default-Einstellungen für ALTER LF
DEFLINES         Default-Einstellungen für ALTER LINES
DEFPROMPT        Default-Einstellungen für ALTER PROMPT
DEFPS            Default-Einstellungen für ALTER PS
DEFREA           Default-Einstellungen für ALTER READLOCK
DEFSTATUS        Default-Einstellungen für ALTER STATUS
DEFUDIR          Default-Einstellungen für ALTER UDIR
DEFULIST         Default-Einstellungen für ALTER ULIST
DEFUREAD         Default-Einstellungen für ALTER UREAD
```

24.5. Initialization file *init.l2*

In der Linux- und in der Windows-Version ist ein AX25-Layer2 enthalten. Dieser wird über die Datei *init.l2* konfiguriert. Hier ein Beispiel:

```
;---------------------------------------------------------------------------
; Parameter-Datei für AX.25-Betrieb der BayCom-Mailbox unter Linux und
; Windows
;
; Stand: 25.12.1998
;
; Änderungen für spätere Versionen möglich
;---------------------------------------------------------------------------
dcall    df0ar-8    ; Digicall (normalerweise unnötig)
monitor  1          ; Monitor, siehe BayCom-Terminal 1.60
mselect  0          ; Dito
mcalls   -0         ; Dito
;---------------------------------------------------------------------------
assign   kiss
mode     9600cd     ; Hier die Baudrate angeben ("c" = RMNC-CRC)
;device  COM1       ; Serielle Schnittstelle: COM1 unter Windows
device   /dev/ttyS0 ; Dito: COM1 = /dev/ttyS0 unter Linux
;---------------------------------------------------------------------------
assign   axip
peer     44.130.1.1 ; Hostname oder IP-Adresse für Verbindung über
                    ; AXIP (Adresse des Zielknotens, nicht die eigene
                    ; Adresse!)
                    ; Achtung: Es handelt sich um AXUDP, "normales" AXIP
                    ; ist nicht möglich
port     4719       ; UDP-Port: Kann beliebig gewählt werden (für
                    ; einen Port unter 1024 sind root-Rechte erforderlich)
;txport  4720       ; Optional: UDP-Port, auf dem gesendet wird, empfangen
                    ; wird immer auf "port"
;---------------------------------------------------------------------------
; Es sind bis zu 20 Ports konfigurierbar...
;---------------------------------------------------------------------------
assign  axip
peer    44.130.1.2
port    4721
txport  4722
;---------------------------------------------------------------------------
```

25. Overview of all mailbox commands

Im Folgenden findet sich eine Übersicht über alle Befehle der OpenBCM-Mailbox. Die Beschreibung ist jeweils auf eine Zeile begrenzt, deshalb ist sie teilweise unvollständig. Soweit die Parameter noch in der Zeile Platz haben, stehen sie

mit dabei, ansonsten wurden sie weggelassen. Parameter in spitzen Klammern müssen angegeben werden, Parameter in eckigen Klammern sind optional. Dabei tauchen folgende Arten auf:

Eine aktuelle Auflistung wird mit dem Befehl CMDLIST bzw. SCMDLIST für Sysopbefehle ausgegeben. Die Verfügbarkeit bestimmter Befehle ist auch von den in der jeweiligen Version eingebauten Optionen abhängig.

<datei>	Für das jeweilige Betriebssystem gültiger Dateiname
<verz>	Für das jeweilige Betriebssystem gültiger Verzeichnisname
<str>	Beliebiger Suchstring, allerdings ohne Wildcards
..	Regulärer Ausdruck, nach dem gesucht wird (RegEx)
<mbox>	Rufzeichen einer Mailbox
<call>	Gültiges Rufzeichen, evtl. mit SSID
<board>	Gültiger Boardname
<addr>	Teil einer hierarchischen Adresse

25.1. Overview of all user commands

#OK#	Wird ignoriert
<GP>:	Wird ignoriert
(BIN-RX):	Wird ignoriert
(BIN-TX):	Wird ignoriert
(YAPP-RX):	Wird ignoriert
(YAPP-TX):	Wird ignoriert
(DIDADIT-RX):	Wird ignoriert
(DIDADIT-TX):	Wird ignoriert
(WPP)	Wird ignoriert
?	Synonym für HELP
AKTUELL	Gibt den Aktuell-Text aus (z.B. *msg/aktuell.dl*)
ALTER	Zeigt die persönlichen Einstellungen an
-AWAY	Abwesenheitstext aktivieren und definieren
-AWAYENDTIME	Abwesenheitszeitraum beschränken
-CHECK	Voreingestellte Optionen beim CHECK-Befehl
-COMMAND	Befehle, die nach dem Login ausgeführt werden
-DEFAULT	Setzt die ALTER-Einstellungen auf Standard-Werte zurück
-DIRFORMAT	Stellt das Format der DIR/READ/LIST-Ausgaben sein
-ECHO	Schickt die eingegebenen Befehle nach der Abarbeitung zurück
-FBBCHECKMODE	Stellt den FBB-Checkmode ein/aus
-FORWARD	Stellt die Heimat-Mailbox ein
-HELPLEVEL	Stufe der Hilfestellung (z.B. Befehls-Menü beim Prompt)
-IDIR	Voreingestellte Optionen beim DIR-Befehl bei Info-Boards
-ILIST	Voreingestellte Optionen beim LIST-Befehl bei Info-Boards
-IREAD	Voreingestellte Optionen beim READ-Befehl bei Info-Boards
-LF	Anzahl der Leerzeilen zwischen den Befehlen
-LINES	Anzahl der Zeilen bis zum Seitenstop
-LINUXPW	setzt das Linux Systempasswort unter Linux
-LOGINPWTYPE	Passwort-Methode beim S&F (BayCom, MD2, etc.)
-NAME	Teilt der Mailbox den eigenen Namen mit
-NEWCALL	Legt bei Lizenzupdate ein neues Rufzeichen fest
-NOTIFICATION	Legt für eine Mailbenachrichtigung ein Rufzeichen fest
-PROMPT	Ändert das Mailbox-Prompt
-PW	Ändert den Passwort-String
-PWLINE	Legt fest, ob vor Passwort-Prompt eine Eingabe erwartet wird
-QTH	QTH einstellen
-READLOCK	Schränkt die Lesbarkeit der eigenen User-Mails ein
-REJECT	Boards, die bei DIR NEWS bzw. CHECK nicht angezeigt werden
-SPEECH	Sprache, in der die Mailbox-Meldungen erscheinen
-UDIR	Voreingestellte Optionen beim DIR-Befehl bei User-Boards
-UFWD	AX.25-Pfad zur eigenen Mailbox (für User-S&F)
-ULIST	Voreingestellte Optionen beim LIST-Befehl bei User-Boards

```
 -UREAD          Voreingestellte Optionen beim READ-Befehl bei User-Boards
 -ZIP            ZIP-Code (=Postleitzahl) einstellen
AUTOPATH <call>  Gibt Informationen vom Autorouter aus
BIDLIST [str]    Gibt alle gespeicherten BIDs aus, die "str" enthalten
BIN-RX           Wird ignoriert
BIN-TX           Wird ignoriert
BYE              Synonym für QUIT
CD <board>       Wechselt in das angegebene Board
CHAT <call>      Synonym für TALK
CHECK [str]      Sucht nach einer Mail bzw. gibt die neuesten Mails aus
CHECKCOUNT       Gibt die Anzahl der neuen Mails seit letztem CHECK aus
CMDLIST          Gibt alle User-Befehle aus
COMMENT          Antwort auf eine Mail in die gleiche Rubrik schreiben
CONVAT           Ausgabe der Datei convat.bcm
CONVERS <call>   Synonym für TALK
CONVERT          Ausgabe der Datei convert.bcm
CP               Synonym für TRANSFER
CRONTAB [..]     Ausgabe der Datei crontab.bcm
DATE             Synonym für TIME
DIR              Listet die Mails in einem Board auf
 -AFTER <Datum>  Listet alle Mails auf, die seit <Datum> eingetroffen sind
 -BOARDS [str]   Listet alle vorhandenen Rubriken auf, die "str" enthalten
 -MESSAGES [str] Listet alle Mails auf, die "str" enthalten
 -NEWS           Listet alle Mails seit dem letzten DIR NEWS auf
 -OUTSTAND [mbox]Zeigt die Mails, die noch zum Partner geschickt werden müssen
 -PATH [mbox]    Zeigt die Ziele, die zu den Forward-Partnern geschickt werden
 -SENT <call>    Listet Bulletins auf, die von einem bestimmten Call stammen
 -USERS          Listet alle Calls auf, für die Mails vorliegen
 --ALL           Listet alle bekannten Calls auf
 --LOCAL         Listet alle Calls auf, die schon mal eingeloggt waren
 --MSG           Listet alle User-Mails auf
ERASE            Löscht eine Mail
ERLOG [..]       Ausgabe der Datei trace/eraselog.bcm
EXIT             Synonym für QUIT
F>               Wechselt in den S&F-Modus
FILESERVER       Synonym für FILESURF
FILESURF         Wechselt in den Filesurf-Modus
FIND <mbox/addr> Synonym für PATH
FINGER <call>    Synonym für USERS
FOLLOWUP         Synonym für COMMENT
FORWARD          Leitet eine oder mehrere Mails an eine andere Mailbox weiter
FS               Synonym für FILESURF
FTPLOG [..]      Ausgabe der Datei trace/ftplog.bcm
FTS [..]         Volltextsuche
FWDCHECK <call>  Gibt Informationen zu einem Forwardziel aus
HEADER           Gibt den Kopf einer Mail aus
HELP             Hilfe (HELP INDEX: Übersicht, HELP ALL: Gesamte Hilfe)
HTTPLOG [..]     Ausgabe der Datei trace/httplog.bcm
IMPDEL           Wird innerhalb einer IMPORT-Datei verwendet
INFO             Gibt einige Konfigurationsdaten und Datei msg/info.* aus
KOPF             Synonym für HEADER
LIST             Synonym für DIR, jedoch ohne Unterbefehle
LOG              Suche in den LOG-Dateien (Verzeichnis log/*.bcm)
LOGOUT           Synonym für QUIT
LT               Synonym für SETLIFETIME
MAILSERVER       Steuert den integrierten Mailing-Listserver
MAN              Synonym für HELP
MD2              Startet die Sysop-Identifizierung mittels MD2-Vefahren
MD5              Startet die Sysop-Identifizierung mittels MD5-Vefahren
MEM              Synonym für STATUS MEMORY
MSG <call>       Synonym für TALK
MYBBS <mbox>     Stellt die eigene Heimat-Mailbox ein (wie ALTER FORWARD)
```

```
NAME              Stellt den eigenen Namen ein (wie ALTER NAME)
NEXT              Gibt die nächste Mail aus (bezogen auf das letzte READ)
NH                Synonym für MYBBS
NNTPLOG [..]      Ausgabe der Datei trace/nntplog.bcm
PARAMETER [..]    Ausgabe der Datei init.bcm
PATH <mbox/addr>  Info über Weiterleitung einer Mail an eine hier. Adresse
PING              Prüft die Erreichbarkeit einer anderen Box
POCSAGSERVER      Steuert den integrierten Pocsag-Listenserver
POP3LOG [..]      Ausgabe der Datei trace/pop3log.bcm
PRIV              Sysop-Identifikation mittels DieBox-Passwort (unsicher!)
PS                Listet alle laufenden Mailbox-Prozesse auf
PURGE             Putzt das eigene Board (als Sysop auch ein anderes Board)
PW                Startet die Sysop-Identifizierung mittels BayCom-Vefahren
PWLOG [..]        Ausgabe der Datei trace/pwlog.bcm
QT                Setzen des Quit-Datums ohne die Box zu verlassen
QUIT              Beendet die Verbindung zur Mailbox
QUOTA             Ausgabe der noch möglichen Bytes an Transfervolumen
READ              Lesen einer Mail
REJECT [..]       Ausgabe der Datei reject.bcm
REPLY             Antwort auf eine Mail an den Absender schreiben
RLOG [..]         Ausgabe der Datei trace/rejlog.bcm
RUNUTILS          Listet die verfügbaren Run-Utilities auf
SA                Verschicken einer ACK-Mail
SB                Verschicken eines Bulletins
SEMAPHORES        Synonym für STATUS SEMAPHORES
SEND              Verschicken einer Mail
SETLIFETIME       Ändern der Lebensdauer einer Mail
SF                Anstoßen des User-S&F (als Sysop auch zum Forward-Partner)
SFHOLD [..]       Ausgabe der Datei trace/sfhold.bcm
SLEEP             Wartet x Sekunden (nur in IMPORT-Dateien sinnvoll)
SLOG [..]         Ausgabe der Datei trace/syslog.bcm
SMTPLOG [..]      Ausgabe der Datei trace/smtplog.bcm
SP                Verschicken einer persönlichen Mail
STATUS            Gibt einige statistische Daten aus
 -CPU             Zeigt die CPU-Auslastung an
 -FORWARD         Gibt die Anzahl der zum Forward anstehenden Mails aus
 -LIMITS          Zeigt Informationen über einige Grenzdaten an
 -MEMORY          Gibt eine Liste der verwendeten Speicherblöcke aus
 -SEMAPHORES      Ausgabe der gesetzten Semaphoren
SYSOP             Synonym für PW
SWAPLOG           Ausgabe der Datei trace/swaplog.bcm
TALK <call>       Schickt einen Text an einen anderen eingeloggten Benutzer
TELL              Einen TELL Befehl zu einer Nachbarbox auslösen
TIME              Gibt das Datum und die (meistens falsche) Uhrzeit aus
TRANSFER          Transferiert eine/mehrere Mails an ein Board/an einen User
UNERASE           Macht eine/mehrere gelöschte Mails wieder sichtbar
UNKNOWN [..]      Ausgabe der Datei trace/unknown.bcm
UNREAD            Mail als ungelesen markieren
USERS             Listet die eingeloggten Benutzer auf
VERSION           Gibt Versions-Nummer und andere Informationen aus
WALL              Schickt einen Text an alle eingeloggten Benutzer
WHOAMI            Für alle, die sich ihr Call nicht merken können...
WRITE <call>      Synonym für TALK
WLOG <call>       Gibt ACTIVE ROUTING Informationen zu einem Ziel aus
```

25.2. Overview of all sysop commands

```
AFWDLIST          Generieren der Autorouter-Datei afwd.bcm
APPEND            Hängt eine Zeile an eine Text-Datei an
BATCH             Startet eine .bat-Datei und die dazugehörige .imp-Datei
BEACON            Sendet die Mail-Bake oder einen beliebigen Text aus
```

BEGIN <datei>	Gibt die ersten zwei kByte einer Text-Datei aus
CALL <call>	Ändert das Login-Call
CAT	Synonym für RTEXT
CFGFLEX	Sendet Konfigurations-Dateien an einen Flexnet-Knoten
CLOG [..]	Ausgabe der Datei *trace/cmdlog.bcm*
CONNECT	Von der Konsole nach außerhalb connecten
CONVEDIT	Editor für *convert.bcm*
DISABLE	Verhindert weitere User- und S&F-Logins
ENABLE	Hebt den DISABLE-Zustand wieder auf
EXPORT	Leitet die Ausgabe eines Befehls in eine Datei um
EXTRACT	Extrahieren von 7+/BIN einer Mail
FEDIT	Editor für *fwd.bcm* (nur DOS, ohne DF3VI_FWD_EDIT)
FWDEDIT	Editor für *fwd.bcm*
FWDEXPORT	Exportieren einer Datei für Datei-Forward
FWDIMPORT	Importieren einer Datei bei Datei-Forward
GREP	Sucht nach einem regulärem Ausdruck in einer Datei
HOLD	Setzt die angegebenen Nachrichten auf HOLD/setzt HOLD zurück
IMPORT	Führt die Zeilen einer Text-Datei als Befehle aus
KILL	Löscht eine Task (Achtung, evtl. unkontrollierbar)
LIFETIME	Setzt die Default-Lifetime für ein Board
LOGIN	Führt einen Mailbox-Login unter dem angegebenen Call aus
MACRO	Führt ein Makro aus (noch nicht fertig!)
MD2SUM <datei>	Berechnet eine MD2 Prüfsumme von <datei>
MD5SUM <datei>	Berechnet eine MD5 Prüfsumme von <datei>
MKBOARD	Richtet ein neues Board ein
MONITOR	Gibt Ein- und Ausgaben einer bestimmten Task aus
MVBOARD	Verschiebt ein Subboard unter ein anderes Hauptboard
NEW	Liest alle Konfigurations-Dateien neu ein
NOTE	Schreibt eine Nachricht ins Syslog
OCP	Führt den internen COPY-Befehl aus
OMD	Führt den internen MKDIR-Befehl aus
OMV	Führt den internen MOVE-Befehl aus
ORM	Führt den internen RM-Befehl aus
OSHELL	Führt einen Befehl auf Shell-Ebene aus
POSTFWD	Trägt alle User-Mails in die Forwardschlange ein
PWGEN <datei>	Erzeugt eine DieBox-Passwort-Datei
RBIN <datei>	Synonym für RPRG
RDIDADIT <datei>	Gibt eine Datei im DIDADIT-Protokoll aus
REDIT	Editor für *reject.bcm* (nur DOS, falls ohne DF3VI_REJ_EDIT)
REJECTEDIT	Editor für *reject.bcm*
REORG	Führt eine Reorganisation der Mailbox-Daten aus
RMBOARD	Löscht ein Board
RPRG	Gibt eine Datei im AutoBIN-Protokoll aus
RTEXT	Gibt eine Text-Datei aus
RYAPP	Gibt eine Datei im YAPP-Protokoll aus
SCMDLIST	Gibt alle Sysop-Befehle aus
SETPW <call>	Ein mit PWG erzeugtes PW kann einem <call> vergeben werden
SETUSER	Ändert die ALTER-Einstellungen eines Benutzers
SHUTDOWN	Beendet die Mailbox
SLR [..]	Ausgabe der Datei *trace/syslog_r.bcm*
TAIL	Gibt die letzten zwei kB einer Text-Datei aus
TEST	Wird des Öfteren zum Testen neuer Funktionen benutzt
TGREP [..]	Ausgabe der Datei *trace/t_<Mailbox-Call>.bcm*
TTYINIT	Initialisieren der mit TTYMODE definierten Schnittstelle
TNC	Ändert eine Einstellung im AX.25-Interface
UIMPORT	Importiert DieBox-Userdaten (*user3.dat* und *user3.idx*)
WBIN	Synonym für WPRG
WDIDADIT	Speichert eine im DIDADIT-Protokoll empfangene Datei
WPRG <datei>	Speichert eine im AutoBIN-Protokoll empfangene Datei
WTEXT <datei>	Speichert eine Text-Datei
WYAPP	Speichert eine im YAPP-Protokoll empfangene Datei

25.3. Overview of all sysop commands at DOS terminal

CEDIT	Lädt die Datei *convert.bcm* in den internen Editor
CONNECT <call>	Versucht, eine AX.25-Verbindung zu einem Call aufzubauen
EDIT <datei>	Lädt eine Datei in den internen Editor
FEDIT	Lädt die Datei *fwd.bcm* in den internen Editor
TWIN	Öffnet ein Fenster mit Trace-Ausgaben
UWIN	Öffnet ein Fenster mit allen eingeloggten Benutzern
W2	Öffnet ein neues Login-Fenster

25.4. Overview of all filesurf commands

#OK#	Wird ignoriert
?	Synonym für HELP
BGET <datei>	Gibt eine Datei im AutoBIN-Protokoll aus
BIN <datei>	Synonym für BGET
BPUT <datei>	Speichert eine im AutoBIN-Protokoll empfangene Datei
BMAIL <datei>	Importiert die Datei als AutoBIN-Mail ins Userpostfach
BYE	Synonym für QUIT
CAT <datei>	Synonym für GET
CD <verz>	Wechselt in ein anderes Verzeichnis
CD..	Wechselt ins übergeordnete Verzeichnis
CHDIR <verz>	Synonym für CD
CMDMODE	Wechselt in den interaktiven Modus
COPY	Kopiert eine Datei
CP	Synonym für COPY
DEL <datei>	Löscht eine Datei
DGET <datei>	Gibt eine Datei im DIDADIT-Protokoll aus
DIR [verz]	Zeigt den Inhalt eines Verzeichnisses an, Wildcards erlaubt
DPUT	Speichert eine im DIDADIT-Protokoll empfangene Datei
EXIT	Synonym für QUIT
GET <datei>	Gibt eine Text-Datei aus
HELP	Gibt die FileSurf-Hilfe aus
LS [verz]	Kurzform des Verzeichnis-Inhalts, mit Parameter "-L" wie DIR
MD [verz]	Legt ein neues Verzeichnis an
MKDIR [verz]	Synonym für MD
MOVE	Verschiebt eine Datei (intern wie COPY + DEL)
MV	Synonym für MOVE
PATH	Zeigt die freigegebenen Verzeichnisse an
PUT <datei>	Speichert eine Text-Datei
QUIT	Beendet den interaktiven Modus
RD <verz>	Löscht ein Verzeichnis
READ <datei>	Synonym für GET
RM <datei>	Synonym für DEL
RMDIR <verz>	Synonym für RD
RPRG <datei>	Synonym für BGET
RTEXT <datei>	Synonym für GET
RYAPP <datei>	Synonym für YGET
TYPE <datei>	Synonym für GET
WPRG <datei>	Synonym für BPUT
WTEXT <datei>	Synonym für PUT
WYAPP <datei>	Synonym für YPUT
YGET <datei>	Gibt eine Datei im YAPP-Protokoll aus
YPUT <datei>	Speichert eine im YAPP-Protokoll empfangene Datei

25.5. Overview of all mailing list server commands

+G	same as NEWGROUP

```
+M              same as ADDMAINTAINER
+U              same as ADDUSER
-G              same as DELGROUP
-M              same as DELMAINTAINER
-U              same as DELUSER
ADDMAINTAINER   add a maintainer to a group
ADDUSER         add a user to a group
DELGROUP        delete a group
DELMAINTAINER   delete a maintainer from a group
DELUSER         delete a user from a group
DESCRIPTION     define a description of a group
HELP            show help for mailserver
INFO            show info of a group
LIST            list all mailserver groups
NEWGROUP        add a new group
OPTIONS         set options of a mailserver group
RESET           reset the mail counter of a group to a value of 0
SETNUMBER       set the mail counter of a group to a special number
SUBSCRIBE       subscribe the own callsign to a group
UNSUBSCRIBE     unsubscribe the own callsign from a group
```

25.6. Overview of all pocsag server commands

```
+G              Synonym für NEWGROUP
+U              Synonym für ADDUSER
-G              Synonym für DELGROUP
-U              Synonym für DELUSER
NEWGROUP        Neue Pocsagserver-Gruppe anlegen
DELGROUP        Ganze Pocsagserver-Gruppe löschen
DESCRIPTION     Beschreibung der Pocsagserver-Gruppe
ADDUSER         User zu einer Pocsagserver-Gruppe hinzufügen
DELUSER         User aus einer Pocsagserver-Gruppe löschen
INFO            Gibt Informationen zu einer Pocsagserver-Gruppe aus
LIST            Listet alle Pocsagserver-Gruppe auf
SUBSCRIBE       Trägt das eigene Call in eine Pocsagserver-Gruppe ein
UNSUBSCRIBE     Trägt das eigene Call aus einer Pocsagserver-Gruppe aus
```

26. Using OpenBCM on citizen band

Die OpenBCM-Mailbox ist für den CB-Funk freigegeben. Support-Anfragen im Bereich CB-Funk zu Einstellungen, Installation, Bugreports, Programmierung und Sonstigem werden von Jonas, DO1MJJ/DJJ812, bearbeitet. Jonas ist auch per Email unter "djj812@dbo812.de" zu erreichen. Ebenso ist die Internetseite http://www.dbo812.de die offizielle Support-Homepage für die CB-Version der OpenBCM-Mailbox.

Um aus der, nach Installation und erstem Start resultierenden Amateurfunkversion der OpenBCM-Mailbox, eine CB-Funk-Version zu generieren, muss der Parameter CALLFORMAT (und evtl. auch LOGINCALLFORMAT) umgestellt werden. Als Parameter wird eine Ziffer verwendet:

    0 = Amateurfunkrufzeichen / Amateurfunkversion der Mailbox
    1 = CB-Rufzeichen / CB-Funk-Version der Mailbox
    2 = Beide Rufzeichengruppen / CB-Funk-Version der Mailbox

Für CB-Funk muss also "CALLFORMAT 1" oder "CALLFORMAT 2" eingetragen werden.

Ist CALLFORMAT auf 1 oder 2 gesetzt, wird auch "(CB)" als Hinweis für die Aktivierung der CB-Version hinter jeder Mailbox-Versionsangabe mit angezeigt.

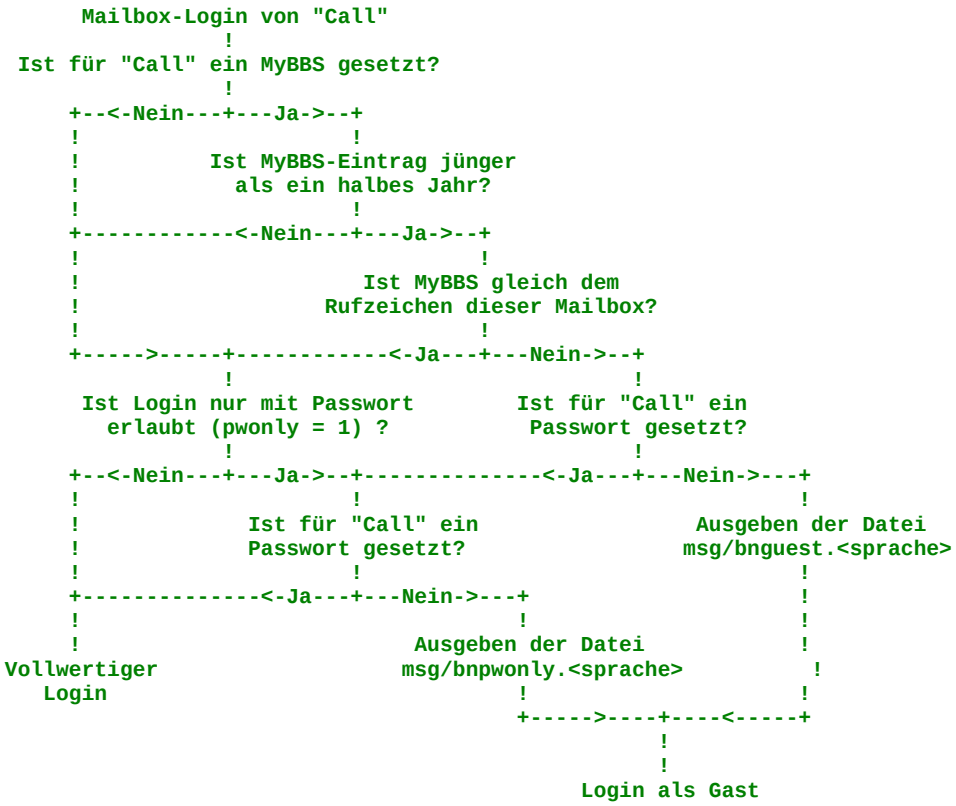
Wurde CALLFORMAT auf 1 gesetzt, so werden folgende Zeichenmuster als gültige CB-Rufzeichen erkannt: XXX###, XX####, XXXX##, XXXXX#, XX##XX, XXX##X, XXXX#X, XXX#XX, XX#XXX, XXX#X, XX#XX, XX##X, ##XX## und XXXX#. Ein X steht dabei für einen Buchstaben und ein # für eine Ziffer. Bei CALLFORMAT 2 sind zusätzlich zu diesen Rufzeichen auch noch alle gültigen Amateurfunkrufzeichenmuster freigegeben.

Generell besteht im CB-Funk zwar keine Rufzeichenpflicht, allerdings dürfen keine bereits vergebenen Rufzeichen, seien es angemeldete CB-Rufzeichen, Amateurfunkrufzeichen oder Rufzeichen irgendeines anderen Funkdienstes, verwendet werden! Generell wird die Anmeldung einer Mailbox auf CB empfohlen, um eine gewisse Organisation zu ermöglichen. Mehr hierzu auf der Homepage der Bundesnetzagentur (ehemals RegTP), unter http://www.bundesnetzagentur.de.

Ferner ist zu beachten, das Flexnet nicht für den CB-Funk freigegeben wurde, somit ist der Benutzung der OpenBCM unter DOS quasi die Grundlage entzogen!

26.1. Function of CB-BCMNET login concept

Im CB-Funk gibt es das Problem, dass die Rufzeichen nicht koordiniert sind und daher mehrfach vorkommen können. Außerdem gibt es Zeitgenossen, welche nichts Besseres zu tun haben als das Netz durch die Benutzung bereits vergebener Rufzeichen durcheinander zu bringen. Das Netz ist damit verwundbar. Aus diesem Grund wurde das CB-BCMNET Loginkonzept mit schaltbarer Sicherheitsstufe (PWONLY in der Datei *init.bcm*) erarbeitet und von Andreas, AS1GBF/DB1RAS, zusammen mit Hannes, AT5HPK, implementiert:



Funktionsprinzip des Loginkonzeptes für das CB-BCMNET:

In eine Mailbox ohne generellen Passwortzugang (PWONLY=0) kann sich nur jeder vollwertig einloggen, wenn

- es sich dabei um ein Rufzeichen handelt, welches in unserem Netz noch nicht bekannt ist, wo also noch kein MyBBS-Eintrag empfangen wurde.
- ein vorhandener MyBBS-Eintrag älter als ein halbes Jahr ist, also davon ausgegangen werden kann, dass dieses Call nicht mehr benutzt wird.
- das MyBBS gleich dieser Mailbox ist, sich der User also in seine Heimatmailbox einloggt.
- auch dann, wenn es sich nicht um die Heimatmailbox des Users handelt, der User aber ein Passwort in dieser Mailbox hat.

Jeder User für den keiner dieser Punkte zutrifft, wird automatisch als Gast eingeloggt. Als Gast hat man vollen Mailboxzugriff, mit folgenden Einschränkungen:

- Alle Mails welche eingespielt werden, bleiben lokal in dieser Mailbox liegen und werden nicht geforwardet, bis der Sysop diese nach Sicht manuell in den Forward stellt.
- Einstellungen wie der Name und die MyBBS werden nicht übernommen, und einige Mailboxbefehle wie erase, transfer,... lassen sich nicht benutzen.
- Alle benutzerdefinierten Einstellungen gelten nur für diesen Login und werden nicht gespeichert.

Bei Mailboxen mit generellem Passwortzugang (PWONLY=1) wird zusätzlich noch geprüft, ob für den User der eines der obigen Punkte erfüllen konnte, ein gültiges Passwort gesetzt ist, ansonsten wird er ebenfalls automatisch als Gast eingeloggt.

Mails welche von einem Gast eingespielt wurden, werden mit

```
X-Info: Mail was sent from a guest
```

im Header gekennzeichnet.

Die Dateien *msg/bnguest.** und *msg/bnpwonly.** werden nur bei dem CB-BCMNET Loginkonzept verwendet. Hier sollten Hinweistexte ausgegeben werden, die auf das CB-BCMNET Loginkonzept und seine Restriktionen hinweist.

27. Compiling the sources

Ist der Quellcode erst einmal ausgepackt (unter Windows kann man zum Beispiel WinZip dafür nutzen, unter Linux langt ein "tar xzf sbcm.tgz" am Prompt), sollte man einen Blick in die Datei *config.h* wagen. Hier kann man spezielle Funktionen der Mailbox aktivieren/deaktivieren.

Die Vorauswahl ist unter Linux bzw. Windows meist optimal (es wird fast alles eingebunden), unter DOS muss man hier jedoch meist eine eigene Auswahl der Optionen treffen. Es ist unter DOS nicht möglich, eine Mailbox-Version zu erstellen, die alle Funktionen beinhaltet, da hierfür der DOS-Hauptspeicher nicht ausreicht. Man muss also abwägen, welche Option man nutzen will, und welche nicht. Zum Deaktivieren einer Option reicht es, ein "//" am Anfang der Zeile einzufügen.

Im Kapitel "27.4. The options of *config.h*" werden alle Optionen der Datei *config.h* erläutert.

27.1. Using DOS

Unter DOS wird der "Borland C++"-Compiler benötigt. Nach Start des Compilers ist die Projektdatei *bcm.prj* zu öffnen. Kommt beim Kompilieren die Meldung "DLINK exceeds > 64k" müssen weitere Optionen in der Datei *config.h* deaktiviert werden.

27.2. Using Linux

Unter Linux muss der GCC/GPP GNU Compiler und seine benötigten Komponenten installiert sein. Dieser findet sich bei allen gängigen Linux-Distributionen.

Im Regelfall reicht dann ein "make" im Source-Verzeichnis, um die Linux-Version der OpenBCM zu kompilieren. Bei "make install" wird dann die erstellte Programmdatei in das */bcm* Verzeichnis kopiert. Existiert dort schon eine Datei */bcm/bcm*, wird von dieser Datei eine Backupdatei */bcm/bcm.old* erstellt. Ändert man Optionen in *config.h*, um diese Auszuprobieren ist es evtl. sinnvoll vor einem neuerlichen "make" ein "make clean" zum Aufräumen des Source-Verzeichnisses zu machen.

Falls beim Kompilieren Warnings (Warnungen) angezeigt werden, kann man diese ignorieren. Treten hingegen Errors (Fehler) auf, so ist meist eine Komponente des GNU Compilers nicht (richtig) installiert. Es ist allerdings auch nicht auszuschließen, dass es sich dann um einen Fehler im Quellcode handelt. Dieser ist aber selten und auf Beta-Versionen beschränkt! ;-)

Es gibt unter Linux noch 3 Spezialfälle:

- bei Verwendung eines alten "SUSE 6.1" Systems sollte man "make -f makefile.suse61" anstelle von "make" aufrufen
- um eine Mailbox ohne Layer2-Support zu erstellen, sollte man "make -f makefile.nol2" anstelle von "make" aufrufen
- um die Mailbox mit dem WX-Modul zu kompilieren (siehe Kapitel "28. WX-Module in Linux version"), ist "make -f makefile.wx" anstelle von "make" aufzurufen

27.3. Using Windows

Unter Windows NT/2000/XP wird die Programmierumgebung "Microsoft Visual C++" mindestens ab Version 5.0 oder höher benötigt. Nach Start der Entwicklungsumgebung ist die Workspace *bcm32.dsw* zu öffnen. Im Compiler ist im Menü "Build/Set active configuration" darauf zu achten, dass man eine "Release"-Version der Mailboxsoftware erstellt. Die "Debug"-Version ist für den Dauerbetrieb nicht ausgelegt und kann zu unschönen Effekten führen!

Hinweis: Seltsamerweise scheint sich die deutsche Compilerversion anders zu verhalten als die internationale (englische) Variante. Mit letzterer gab es bislang keine Probleme, während eine Mailbox, die mit der deutschen Variante kompiliert wurde manchmal im Betrieb zu seltsamen Abstürzen neigt.

27.4. The options of *config.h*

Generelle Optionen:

ALPHA	Der Hinweis "alpha"-Version wird bei der Versionskennung eingeblendet
RUNUTILS	Runutil-Funktion
MAILSERVER	Mailing-Listserver
FILESURF	Filesurf
DF3VI_POCSAG	PocSag-Message-Server
FEATURE_YAPP	Unterstützung für das YAPP-Übertragungsprotokoll
FEATURE_MDPW	Unterstützung für MD2/MD5 Passwörter
DF3VI_FWD_EDIT	Editorfunktionen für *fwd.bcm* (nur für Sysops)
DF3VI_REJ_EDIT	Editorfunktionen für *reject.bcm* (nur für Sysops)
DF3VI_CONV_EDIT	Editorfunktionen für *convert.bcm* (nur für Sysops)
DF3VI_EXTRACT	Extrahieren von 7+/AutoBIN-Teilen aus Mails mit dem Befehl EXTRACT (nur für Sysops)
USERLT	Die Lifetime des Absenders einer Mail wird angezeigt
_AUTOFWD	Automatischer Forward-Router
_GUEST	Gastrufzeichen "GUESTCALL" für TCP/IP und TTY Zugriff
_FILEFWD	Dateiforward-Unterstützung, z.B. für Email-Forward
_LCF	Unterschiedliches Callformat für Userlogin und Forward
DIEBOX_UIMPORT	Importiert Userdaten aus DieBox Datei *user3.dat* und *user3.idx*, wird normalerweise nicht benötigt
_USERS4CONVERT	Modul zur automatischen Konvertierung der Userdatenbank vom alten Format *users.bcm/users3.bcm* ins neue Format *users4.bcm*
FBBCHECKREAD	Gibt bei CHECK eine Liste mit Mailnummern wie in FBB aus, der READ-Befehl versteht dann auch ein "R <Nr>"
MACRO	Makro-Sprache
FULLTEXTSEARCH	Volltextsuche (Befehl FTS)
FEATURE_DIDADIT	Unterstützung für das DIDADIT-Übertragungsprotokoll
OLDTIMEFMT	Verwendet Zeitformat in Log-Dateien im Format der BCM v1.42, normalerweise nicht mehr sinnvoll

Optionen nur für DOS:

SMALL_DOS	Aktiviert eine kleine DOS-Auswahl am Ende von *config.h*
_TNC	Unterstützung für TNC/Flexnet
_TELEPHONE	Telefon-Forward
_AUTOTRCWIN	(Auto)Trace-Fenster
FEATURE_EDITOR	Interner Editor
FEATURE_COLOR	Farbauswahl mit ALT-C
FEATURE_MOUSE	Mausunterstützung
BIOS_Y2K	Umgeht Jahr2000-Probleme von alten BIOS-Versionen
OLD_SHELL	Aktiviert die alte DOS-Shell (nicht sinnvoll)
_PREPOSTEXEC	Externer DOS Watchdog (Befehle DOSPREEXEC, DOSPOSTEXEC)

Optionen für Linux und Windows:

NNTP	NNTP-Server
SERVIF	Simples Telnet Service-Interface
_TELNETFWD	Telnet-Forward

Optionen nur für Linux:

FEATURE_SERIAL	Unterstützung für TTY-Login
RADIOIF	Radio Interface (Net-CMD TNT/Wampes)
LINUXSYSTEMUSER	Aktiviert ALTER LINUXPW, um Systemlogins zu erzeugen
_AX25K	Unterstützung des Kernel AX.25

spezielle Debug-Ausgaben (nur zur Fehlersuche sinnvoll):

DEBUG_DIDADIT	Debugausgaben für DIDADIT-Dateitransfer
HB9EAS_DEBUG	Debugausgaben für ACTIVE ROUTING-Tests von DB0FHN+HB9EAS
DEBUG_AFWD	Debugausgaben für Autofwd
DEBUG_FWD	Debugausgaben für Forward allgemein
DEBUG_FTP	Debugausgaben für FTP
DEBUG_HTTP	Debugausgaben für HTTP
DEBUG_POP3	Debugausgaben für POP3
DEBUG_NNTP	Debugausgaben für NNTP
DEBUG_SMTP	Debugausgaben für SMTP
_DEBUG_SEMA	Debugausgaben für "NO_SEMA debugging"

spezielle Optionen:

_WXSTN	Wetterstation von Peetbros (Ulitimiter 2000)
_BCMNET	aktiviert Funktionen für das BCMNET im CB-Funk

Beta-Optionen:
Werden hier nicht näher erläutert, da sie sich häufig ändern können!

27.5. Look backwards: one common source

Die hohe Kunst der portablen Programmierung besteht nicht darin, möglichst verschiedene Versionen für verschiedene Betriebssysteme zu erzeugen, sondern möglichst eine einheitliche Software zu pflegen, die dann auf allen Plattformen übersetzbar und lauffähig ist.

Wird dieses Ziel gut verwirklicht, dann ist auch die Portierung auf weitere Plattformen ohne größere Schwierigkeiten möglich.

So erklärt sich auch, warum es stets nur eine Version gibt und zu einer bestimmten Versionsnummer auch eine bestimmte Funktionalität gehört, ohne dass für die jeweils andere Plattform zusätzliche Arbeit nötig ist.

Trotz dieser guten Vorsätze waren natürlich durchaus Änderungen erforderlich, um den Code portabel zu machen. Folgende Dinge müssen dabei beachtet werden:

- DOS-Dateinamen dürfen groß- oder kleingeschrieben werden. Bei Linux ist die Schreibweise relevant und sollte generell klein sein.
- Integer-Variablen haben unter DOS 16 Bit Darstellungsbreite, bei allen anderen Betriebssystemen sind 32 Bit üblich. Im Einzelfall dürfen keine Annahmen über die Darstellungsbreite getroffen werden. Notfalls müssen Typen definiert werden, die in systemabhängigen Headerdateien definiert werden.
- Systemaufrufe verhalten sich teilweise anders. Sie müssen deshalb in betriebsystemabhängige Module gekapselt werden, um in Richtung zur Anwendung identisch zu sein.
- Hardwarezugriffe müssen vermieden werden. Wenn unbedingt erforderlich, ist eine Virtualisierung sinnvoll, so dass die Anwendung nichts davon sieht.
- Die Programmiersprache muss einheitlich sein. Assemblerteile sind möglichst zu vermeiden, weil diese gewiss nicht portabel sind.

So ergibt sich eine kleine Zahl von Modulen, in denen Abhängigkeiten vom Betriebssystem enthalten sind. Der gesamte Rest (>95% der Gesamtmenge) enthält weder Abhängigkeiten vom Betriebssystem noch von der Hardware.

27.5.1. The same source for all operating systems

Für den Benutzer ergeben sich kaum Unterschiede zwischen den Versionen. Eine wesentliche Zielsetzung bei der Entwicklung der OpenBCM-Mailbox war schon in der DOS-Version, dass für den Benutzer stets eine gewohnte Umgebung mit vorhersehbarem Verhalten existiert. Waren es bei der DOS-Version andere Mailboxsysteme, die als Vorbild dienten, so galt es bei der Linux- und Windows-Portierung, möglichst viele der bisherigen Bedienungselemente exakt beizubehalten und nicht einzuschränken oder zu verändern.

Diese Zielvorgabe ist auch weitestgehend gelungen, so dass auf der Benutzerseite tatsächlich so gut wie keine Änderungen erkennbar sind. Dies gilt allerdings nur für den tatsächlichen "Benutzer" über Funk. Sysops haben mit sehr wesentlichen Unterschieden, die im Kern des jeweiligen Betriebssystems zu suchen sind, zu rechnen.

27.5.2. Some problems could be easy solved

Natürlich gibt es einige Programmteile, die sich partout nicht so anpassen lassen, dass sie für alle Plattformen gemeinsam nutzbar sind. Hier mussten echte "Sonderlocken" entwickelt werden, die auch bei zukünftigen Portierungen wieder neu gemacht werden müssen.

27.5.2.1. Exit to operating system (sysop shell command and run utilities)

Bei DOS ist ein Shell-Ausstieg durch Nachladen von *command.com* möglich. Während eines solchen Ausstiegs steht das aufrufende Programm still. Um einen Ausstieg über Funk möglich zu machen, müssen vorher die Ein- und Ausgaben über die BIOS-Schnittstellen umgelenkt werden. Dies ist kritisch und erfordert sehr systemnahe Eingriffe. Unter Linux erfolgt ein Ausstieg ganz anders und vor allem sehr viel eleganter. Wichtigster Vorteil ist, dass während eines shell-Ausstiegs die Box ohne Behinderung weiterläuft. Die Kommunikation zwischen dem Boxprogramm und der shell (bzw. des ansonsten ausgeführten Programms) erfolgt über ein Pseudo-TTY. Nach Beendigung des aufgerufenen Programms müssen alle davon erzeugten Prozesse wieder sauber aufgeräumt werden.

27.5.2.2. Task switching in multitasking scheduler

DOS kann bekanntlich nur ein Programm gleichzeitig abarbeiten (von Interrupts abgesehen). Damit die Box viele Benutzer gleichzeitig bedienen kann wurde für die DOS-Version ein Multitasking-Kern entwickelt, der die Anwendung selbst von dieser Aufgabe befreit. Jeder eingelegte Benutzer erhält eine eigene Umgebung und muss sich zum eigenen Ablauf nicht um die Abläufe anderer Benutzer kümmern.

Dieses Subsystem wäre in dieser Art unter anderen Betriebssystemen nicht erforderlich, wenn sog. Threads, also verschiedene Tasks innerhalb einer Anwendung vom Betriebssystem zur Verfügung gestellt werden. Unter Linux wäre dies im Prinzip möglich. Um das Programm so gleich wie möglich zu halten, werden jedoch auch hier die Mechanismen der DOS-Version benutzt. Dies bedingt, neben vielen portabel realisierbaren Mechanismen, die Umschaltung des Stack-Kontextes. Zu diesem Zweck ist genau ein Assemblerbefehl notwendig, der ein Verbiegen des

Stackpointers möglich macht. Dieser Befehl ist sowohl von der Hardware, als auch vom Compiler, als auch vom Betriebssystem abhängig und muss deshalb jedes Mal überarbeitet werden.

27.5.2.3. Packet radio accessibility

Die DOS-Version kommuniziert über einen Software-Interrupt mit Flexnet. Dabei sind nur kompakte Interfaceroutinen in der Box eingebaut, das gesamte AX.25-Handling wird im Interrupt von einem TSR-Programm durchgeführt.

In der Linux-Version ist derzeit der AX.25-Teil mit in der Box enthalten. Dieser kommuniziert nach außen nur über vorhandene Standard-Schnittstellen. Konkret sind dies CRC-KISS-Mode über serielle Schnittstelle sowie AXIP über UDP für Ethernet-Kommunikation. Außerdem kann die Kommunikation nach außen über ein Linux-Kernel-AX.25-Interface erfolgen.

27.5.2.4. Screen/Keyboard

Bei DOS wird direkt in den Bildschirmspeicher geschrieben. Dies ist notwendig, um eine ansprechende Konsole zur Verfügung zu stellen. Es gibt nur diesen einen Bildschirm, er muss deshalb alle Erfordernisse zur Wartung der Box zur Verfügung stellen. Die Linux-Version wird ganz im Hintergrund betrieben, es findet also keinerlei Bildschirmzugriff statt. Dafür ist es bei Linux vom Betriebssystem möglich, gleichzeitig mehrere virtuelle Bildschirme zu betreiben und von dort über Interprozesskommunikation in die Box einzuloggen.

27.5.2.5.Timer interrupt (Generation of a signal)

Bei DOS muss zu diesem Zweck ein Interrupt-Vektor auf das eigene Programm gelegt werden.

Bei Linux wird ein zyklisches ALARM-Signal erzeugt, das dann ebenfalls einen Interrupt im Anwendungsprogramm auslöst. Der Zyklus wurde hierbei auf die bei DOS üblichen 55ms festgelegt.

28. WX-Module in Linux version

In diesem Kapitel wird beschrieben, wie eine WX-Station vom Typ "Peetbros Ultimeter 2000" (siehe http://www.peetbros.com) mit einer Linux-OpenBCM betrieben werden kann. Das WX Modul wurde 1998 von OE3DZW für die Linux-OpenBCM entwickelt. Das Modul kann nicht unter DOS oder Windows verwendet werden!

28.1. Compilation of WX-Module for usage

Um die WX-Funktion in die Mailbox einzubinden, muss das *makefile* vor dem Kompilieren angepasst werden. Es muss lediglich die Zeile

```
LD_OPT += -lm
```

mit eingebunden werden. Danach wie gewohnt mit "make install" kompilieren. Während des Kompilierens erscheinen evtl. ein paar Kompiler-Warnungen wegen *mbwx.cpp*, diese können aber ignoriert werden.

28.2. Connection of WX-Module

Die WX Station sollte über einen seriellen Port (z.B. COM1 /dev/ttyS0) mit 2400 Baud an den PC angebunden sein.

Das WX-Modul benötigt ein zusätzliches Unterverzeichnis, z.B. */bcm/wxdata*. Um dies zu erstellen, sollte man eingeben:

```
mkdir /bcm/wxdata
chown bcm:bcm /bcm/wxdata
```

28.3. Configuration of WX-Module

Nach dem Start der OpenBCM mit einkompiliertem WX-Modul sollte man sich als Sysop in die Mailbox einloggen. Folgende zusätzliche Parameter können nun für das WX-Modul eingestellt werden:

```
wxtty /dev/ttyS0      # für seriellen Port COM1
wxpath /bcm/wxdata    # für den Pfad zu "/bcm/wxdata"
wxstnname <name>      # max. 30 Zeichen
wxqthaltitude <n>     # <n> in Metern
wxsensoraltitude <n>  # <n> in Metern
```

29. OpenBCM script language

Sowohl in der Windows, als auch in der Linux-Version ist eine Makrosprache implementiert (BCSL = BayCom Script Language). In der DOS-Version ist diese Makrosprache nicht vorgesehen (zu wenig Speicher).

Die Makrosprache ist bis jetzt noch eine Baustelle.

!!! Vorsicht. Diese Beschreibung ist unvollständig und in Teilen falsch! Auch die Software ist noch unvollständig und ungetestet!!!

Dieses Kapitel dient lediglich der Sammlung von Ideen.

29.1. Introduction

Es wird oft gewünscht, in eine vorhandene Mailbox zusätzliche Funktionen einzubinden. Eine gewisse Möglichkeit ist durch die Einbindung von Run-Utilities gegeben. Diese sind allerdings sehr unflexibel und können nur stark eingeschränkt auf die Datenbestände der Mailbox zugreifen.

Um diese Lücke zu schließen, wurde eine Skriptsprache entwickelt, mit der kleine Zusatzfunktionen realisiert werden können, für die das Mailboxprogramm nicht verändert werden muss. Was dabei herausgekommen ist, wird jedem Informatiker, der sich bereits mit formalen Sprachen beschäftigt hat, die Haare zu Berge stehen lassen. Beruhigend ist jedoch die Tatsache, dass es relativ viele Skriptsprachen gibt, die ebenso wirr strukturiert sind und dabei einen weit größeren Verbreitungsgrat haben als BCSL.

29.2. Executing a script

Ein Skript kann zu Testzwecken vom Sysop per Befehl gestartet werden. Dies geht mit "macro <scriptname>". Die Skripte stehen in einem Verzeichnis namens *macro*. Jedes Skript wird in einem eigenen File abgelegt. Der Name des Skriptes ist der Dateiname (ohne Endung, bzw. mit Endung ist auch diese Bestandteil des Scriptnamens) Im praktischen Betrieb erfolgt die Abarbeitung von Skripten ereignisgesteuert. Immer wenn ein Ereignis unter bestimmten Randbedingungen

eintritt, kann ein bestimmtes Skript ausgeführt werden. Die Verbindung zwischen Ereignissen und Skriptnamen wird in einem File namens *macro.bcm* festgelegt.

Dieses File hat Einträge im folgenden Format:

```
<event> [<call> | <command> | <wildcard>]: <macro>
```

Folgendes Beispiel zeigt grob die Möglichkeiten:

```
login *: loginout
login dg3rbu: rbu
accept db0aab-3: aab3
cmd a*ktuell: akt
cmd pu*rge: disabled
scmd fwdadd: fwdadd
beforecmd v*ersion: b
beforecmd *:
aftercmd *:
mailfm dh3vy:
deletemail mailto *:
unknown db0kfb*: nobox
```

In der ersten Spalte wird das Ereignis benannt, das zur Ausführung des Skriptes führen wird. Hinter dieser Angabe erfolgt ein Parameter, der das Ereignis entweder konkretisiert oder einschränkt. Abkürzungen mit * sind meist möglich, die alleinige Angabe von * ist grundsätzlich zutreffend und ist gleichwertig mit dem Weglassen dieses Parameters, d.h. "login *: script" und "login: script" verhalten sich gleich.

Hier gibt es folgende Möglichkeiten:

login [<call>]: Nach jedem Login wird dieses Makro ausgeführt. Der Zeitpunkt der Ausführung ist, nachdem alle Loginformalitäten erledigt sind aber noch bevor das erste Kommando (siehe "alter command") ausgeführt wurde. Durch einen Parameter kann das Rufzeichen des Logins eingeschränkt werden. Mittels * ist das Rufzeichen abzukürzen. Denkbar wäre z.B. ein Konstrukt: "login oe*: oegruss", wenn im Skript "oegruss" alle Freunde aus unserem südöstlichen Nachbarland besonders herzlich begrüßt werden.

accept <call>: Hier wird ein Connect von außen unter dem angegebenen Rufzeichen entgegengenommen. Der Connect wird entgegengenommen, die Kommunikation mit dem neu ankommenden Benutzer wird direkt an das angegebene Skript übergeben.

Wird das Skript beendet, so erfolgt ein Disconnect. Bei Logins über Telnet oder am Bildschirm kann das Rufzeichen als zweiter Parameter beim Login angegeben werden. An dieser Stelle ist eine Abkürzung mit * nicht zulässig.

cmd <command>: Fügt ein neues Kommando der Boxoberfläche hinzu. Das Kommando kann mittels * in einen verbindlichen und einen optionalen Teil zerfallen.

Beispiel:
new*command bedeutet, dass ein neuer Befehl "newcommand" eingefügt wird, wobei "new" in jedem Fall eingegeben werden muss, der Rest kann entfallen. So eingeführte Befehle können eingebaute Befehle der Mailbox überschreiben, diese sind dann mit dem Original-Syntax nicht mehr ausführbar.

Durch Voranstellen eines "-"Zeichen wird jedoch die Ausführung des eingebauten Befehles erzwungen, so dass ein tatsächliches Abdecken eines Befehles nicht möglich ist (dies ist kein Bug sondern Absicht).

scmd <command>: wie "cmd", jedoch sind so erzeugte Kommandos nur im Sysop-Modus ausführbar.

beforecmd <cmd> und **aftercmd <cmd>**: Hier referenzierte Skripte werden vor bzw. nach der Ausführung des mit <cmd> referenzierten Kommandos ausgeführt.

Die Abkürzung des Kommandos muss genauso aufgeschrieben werden, wie dieses von der Box interpretiert wird.

mailfrom <call> und **mailto <call>**: Dieses Skript wird aufgerufen, wenn eine neue Nachricht in der Box eintrifft. Über den Parameter kann ggf. eingeschränkt werden, zu wem (bzw. in welche Rubrik) oder von wem eine Nachricht eingespeichert wurde. Beim Forwarding ist dabei das tatsächliche Absenderrufzeichen der Mail relevant, nicht das Loginrufzeichen der Forwardbox. Wird kein Parameter oder * angegeben, sind die Ereignisse "mailto" und "mailfrom" identisch.

unknown <adr>: Das referenzierte Skript wird ausgeführt, wenn eine Nachricht mit einer unbekannten Forwardadresse eintrifft. Bei der Adresse ist eine Abkürzung mit * möglich.

29.3. Syntax elements of the script language

Die BCM-Makroskriptsprache ist eine frei erfundene Programmiersprache, die zwar sehr viele Gemeinsamkeiten mit anderen Sprachen hat, zu keiner jedoch identisch ist. Der Kritiker mag dies schimpfend bemängeln, wird es aber auch nicht mehr ändern können. Als Begründung, warum die Sprache so aussieht und nicht anders kann man lediglich anführen, dass sie eben so geworden ist. Eine belastbare technische Begründung gibt es nicht. Wesentlich ist die Tatsache, dass es nicht möglich ist, komplexe Programme in BCSL zu schreiben.

Um keine falschen Hoffnungen zu wecken, seien hier zunächst einmal die Schwächen von BCSL aufgezählt:

- mit BCSL sind nur überschaubare, oberflächliche Problemstellungen lösbar.
- Es existieren keine strukturierten oder indizierten Datentypen.
- Unterprogramme sind nur mit gewissen Einschränkungen möglich.
- Durch textuelle Interpretation ist die Ausführung der Skripten ausgesprochen langsam.
- Es ist nur Integerarithmetik mit den Grundrechenarten vorhanden.
- Durch relativ hohen Speicherbedarf ist die Funktion in der DOS-Version normalerweise gar nicht, in Sonderfällen nur eingeschränkt nutzbar.

Jedes Skript wird durch die Schlüsselwörter "begin" und "end" eingeklammert. Auch Kontrollstrukturen ("if" und "loop") werden gezielt durch "endif" und "endloop" beendet. Jedes Statement wird mit ";" abgeschlossen. Es gibt einen Unterschied zwischen Prozeduren und Funktionen. Funktionen haben einen Rückgabewert, dieser muss auch weiterverarbeitet werden (z.B. in einem Klammerausdruck oder einer Zuweisung). Prozeduren haben keinen Rückgabewert und können nicht zugewiesen werden. Das Verhalten in dieser Beziehung ist ähnlich Pascal. Innerhalb einer Blockstruktur (zwischen "begin" und "end" oder sinngemäß innerhalb einer Kontrollstruktur "if" oder "loop") wird eine Serie von Statements ausgeführt, die jeweils durch ";" abgeschlossen werden.

29.4. A script in principle

Der prinzipielle Aufbau eines Skripts erfolgt nach folgendem Muster:

```
begin <script-Block>
    if( Bedingung )
        <script-Block>
    else
        <script-Block>
    endif;
    loop
        <script-Block>
        exit_if( Bedingung );
```

```
            <script-Block>
        endloop;
        <script-Block>
    end.
```

Example:

```
    // Script for output of "Aktuell"-Text
    begin
        ftime=filetime("msg/aktuell.dl");
        if(ftime!=0) put("Aktuell vom " & datestr(ftime) & " " & timestr(ftime)
             & "\n");
          cmd("rt msg/aktuell.dl");
        else put("Kein aktueller Text geladen.\n");
        endif;
    end.
```

29.5. Comments

Im Skript dürfen an beliebigen Stellen Kommentare stehen. Die Konventionen entsprechen hier der Sprache C++:

- Die Zeichen /* und */ schließen einen Kommentar innerhalb einer Zeile ein. Hierbei ist keine Schachtelung zulässig.
- Das Zeichen // leitet einen Kommentar bis zum Ende der Zeile ein.
- Kommentare dürfen an beliebigen Stellen im Quelltext stehen, jedoch nicht innerhalb eines Bezeichners (Konstante, Variable, Schlüsselwort etc).

29.6. Variables

Variablen werden implizit deklariert. Das heißt, dass eine Variable als deklariert und definiert gilt, sobald sie die linke Seite einer Zuweisung passiert hat. Eine Variable kann ohne Typdeklaration sowohl numerischen als auch alphanumerischen Inhalt annehmen, auch eine Zuweisung von einem Typ zum anderen ist zulässig. Eine leere (aber definierte) Variable kann durch x=""; erzeugt werden. Der Inhalt einer Variablen kann beliebig lang werden (Begrenzung durch die Ressourcen des Rechners, bei DOS 64kByte), numerische Werte reichen von -2^31 bis 2^31 (32 Bit Integer).

29.7. Constants

Im Programmtext werden Stringkonstanten in "Anführungszeichen" angegeben. Bei numerischen Konstanten ist sowohl eine Angabe mit als auch ohne Anführungszeichen zulässig. Innerhalb von Stringkonstanten können die C-üblichen Steuerzeichen verwendet werden, das sind:

- \n Newline, also Sprung an den Anfang der nächsten Zeile
- \a Klingelzeichen, also CTRL-G
- \" Anführungszeichen "
- \xhh Hexadezimale Angabe des Zeichencodes, der eingesetzt werden soll. Die Angabe wird stets 2stellig erwartet, z.B. \x00 oder \xff Beispiele: x=10; y="das ist ein String mit Zeilenende\n"; xx=27+y; ist gleichwertig zu xx="27"+y;

29.8. Reserved words

Die Schlüsselwörter von BCSL teilen sich auf in:

- Operatoren
- Strukturanweisungen
- interne Prozeduren
- interne Funktionen

Operatoren (Auflistung in der Reihenfolge der Priorität):
Bei allen arithmetischen Operationen werden leere Variablen, oder Variablen die keine Zahl beinhalten, zu 0 angenommen (eine solche Operation führt nicht zum Fehler). Bei logischen Operationen gilt der Wert 0 (oder keine Zahl) als "FALSE", jede Zahl ungleich 0 gilt als "TRUE". Zurückgegeben wird "0" für "FALSE" und "1" für "TRUE". Das Verhalten in dieser Beziehung ist gleich der Sprache C.

Für Variablen, die keine Zahlen enthalten (also Strings) sind nur die Operatoren "==", "!=" und "&" sinnvoll einsetzbar.

- Punktoperatoren:
  * multipliziert numerische Ausdrücke
  / dividiert numerische Ausdrücke
  mod Modulo-Operation (Rest der Division)
- Strichoperatoren:
  + addiert numerische Ausdrücke
  - subtrahiert numerische Ausdrücke
- Vergleichsoperatoren:
  > größer als
  < kleiner als
  >= größer oder gleich
  <= kleiner oder gleich
  == gleich (bei Strings: identisch, Groß-/Kleinschreibung irrelevant)
  != ungleich (auch für Strings anwendbar)
- logische Operatoren
  0 oder
  and logisches UND
  or logisches ODER
  xor exklusives ODER
- Stringverkettung:
  & fügt Strings aneinander

29.9. Structure assignment

- begin und end umklammern ein ganzes Skript.
- Jedes Skript wird durch die Folge begin <block> end; eingeschlossen.
- if, else und endif machen eine Verzweigung nach einer Bedingung möglich. Das Verhalten ist mit jeder anderen strukturierten Programmiersprache vergleichbar.
- Eine Schachtelung solcher Strukturen ist möglich und hat nicht das von C bekannte Problem der Uneindeutigkeit von "else", da jede Struktur mit "endif" eingeklammert wird.
  if (<Bedingung>) <block> endif; oder
  if (<Bedingung>) <block> else <block> endif;
  Der Ausdruck <Bedingung> wird wie in C numerisch ausgewertet. Jeder i numerische Wert ungleich 0 ist "true", 0 ist "false".
  Ein nichtnumerischer Wert wird ebenfalls als "false" interpretiert.
- loop, exit, exit_if, endloop: Dies sind die Elemente des Schleifenkonstruktes. Ein solches Konstrukt ist z.B. aus der Sprache ADA bekannt und ersetzt die verschiedenen Konstrukte wie "do ... while" oder "repeat ... until". Auch eine FOR-Schleife existiert hier nicht.
  loop <block> exit_if (<Bedingung>);
  <block> endloop;
  Der Block vor oder nach dem "exit_if"-Ausdruck kann jeweils auch entfallen, dadurch erfolgt die Überprüfung der Ausstiegsbedingung vor oder nach dem Schleifendurchlauf. Die Bedingung wird wie bei "if" ausgewertet. Das Schlüsselwort "exit" bewirkt einen harten Ausstieg aus der Schleife ohne Bedingung, dies wird jedoch wohl eher selten benötigt.

29.10. Internal procedures

- abort(<reason>); bricht ein Makro hart ab.
- call(<macroname>[, <var1>,...]); ruft ein Unterprogramm auf. Dabei können Variablen für das Unterprogramm sichtbar gemacht werden.
- disconnect(<handle>); geht noch nicht.
- filewrite(<filename>, <data>, <mode>); schreibt eine Variable in ein File.
- oshell(<command>); führt einen shell-Befehl aus.
- put(<string>); gibt einen String zum Benutzer aus.
- sleep(<milliseconds>); suspendiert das Skript.
- trace(<level>, <string>); gibt einen Trace ins Syslog aus.
- cmd("e dl8mbt 1-"); führt ein Boxkommando aus. Ausgaben erfolgen zum Benutzer.

29.11. Internal functions

- x=chr(65); * one ASCII-character erzeugt ein Zeichen gemäß angegebenem Code.
- x=cmd("v"); führt ein Kommando aus und weist die Ausgabe des Kommandos einer Variable zu.
- handle=connect("dl8mbt", "db0aab"); geht noch nicht.
- x=datestr(gettime); wandelt ANSI-Zeit in lesbaren String um.
- line=fileline("filename",5); geht noch nicht.
- string=fileread("filename"); liest ein File ein und weist es einer Variablen zu.
- size=filesize("filename"); gibt die Größe einer Datei zurück.
- t=filetime("filename"); gibt den Zeitstempel einer Datei als ANSI-Zeit zurück.
- s=getfname("dl8mbt",25); geht noch nicht.
- s=getline; geht noch nicht. s=getlist("dl8mbt"); geht noch nicht.
- s=getmacro("%o"); geht noch nicht.
- t=gettime; gibt die momentane ANSI-Zeit zurück.
- s=getuser("dl8mbt","forward"); gibt eine ALTER-Einstellung eines Users zurück. Folgende Einstellungen können abgefragt werden:
  check command echo fdelay fhold forward grep helplevel idir ilist iread lf lines mybbs name nopurge prompt pw pwline quota readlock reject rlimit speech status ttypw udir ulist uread
  ; geht noch nicht.
- boolean=getvalid; geht noch nicht.
- call=getvar("logincall"); gibt den Inhalt einer System-Variablen zurück. Folgende Bezeichner können übergeben werden:
  logincall uplink txbytes rxbytes sysop logintime board dest src frombox at bid boxlist usermail lifetime lines bytes subject replyboard replynum taskid taskname created lastinput lastcmd
  ; geht noch nicht.
- i=len(string); gibt die Länge eines Strings zurück.
- a=lines(x); gibt die Anzahl der Zeilen in einem String zurück, geht noch nicht.
- x=oshell("ls -l"); or oshell("ls -l"); shell-Ausstieg. Zuweisung geht noch nicht.
- s=strline(x,5); gibt die 5-te Zeile eines Strings zurück, geht noch nicht.

- i=strpos(s,"test"); gibt die Position eines Substrings in einem String zurück.
- t=timestr(time); wandelt ANSI-Zeit in Zeitstring um
- s1=token(command,1); zerlegt einen String in einzelne Teile, geht noch nicht.
- x=val("A"); gibt den Zeichencode des ersten Zeichens im String zurück

Wichtig: Dieses Kapitel dient lediglich der Sammlung von Ideen!

30. Extended White Pages Protocol (WPROT) specification v1.0/Rev.2

Dieses Kapitel enthält eine Beschreibung des WPROT Protokolls, einem Verfahren für den Austausch von "White Page"-Informationen zwischen Packet-Radio Mailboxen. WPROT basiert auf den älteren Standards von W0RLI und F6FBB, die allgemein als "WP"-Verfahren bekannt sind. Diese Spezifikation wurde ursprünglich von OE3DZW entwickelt und greift auch Ideen zum Routing-Informationsaustausch von DL8HBS auf.

30.1. Functional overview

WPROT ist...

- hocheffizient
- bietet End-to-End-Fehlerkontrolle
- hochflexibel
- skalierbar
- Sicherung der Datenübertragung durch Checksumme

30.2. Compatibility

WPROT ist abwärtskompatibel zum WP-Standard von W0RLI und zum E/M-System wie es von Diebox genutzt wird. /G und /I (guess and info) Nachrichten von WP werden allerdings nicht unterstützt.

30.3. Why a new protocol?

Das originale WP Protokoll unterstützt nur den Austausch von Heimatmailbox-Informationen. Der Austausch dieser Daten ist jedoch unsicher (jeder User kann irgendwelche WP Nachrichten absenden), der Zeitstempel beinhaltet nicht den Zeitpunkt des Tages und es ist nicht kompatibel mit dem E/M-System von Diebox, dass immer noch von zahlreichen anderen Boxsystemen genutzt wird. Die Y2K-Kompatibilität ist bei älteren Umsetzungen des WP Protokolls nicht gegeben (Datum im "990407"-Format).

Das E/M-Protokoll von DieBox bietet ebenfalls Nachteile: es ist sehr uneffektiv, denn nur eine Information kann pro Mail versendet werden. Es unterstützt zwar Remote-Erase (Fernlöschung) von Rubriknachrichten, dies ist technisch jedoch nicht bis ins letzte Detail gut implementiert - somit konnte dies leider in der Vergangenheit von Löschsysops massiv missbraucht werden.

WPROT ist hingegen mit E/M und WP kompatibel, nutzt die Vorteile von beiden Protokollen und bietet eine zusätzliche Funktionalität.

30.4. Activating WPROT

Der Sysop hat lediglich sicherzustellen, das WPROT Daten zu WPROT-kompatiblen Nachbarboxen gesendet werden. In OpenBCM geschieht dies, indem im Forwardabschnitt zur Nachbarmailbox das Schlüsselwort $WP in die Datei *fwd.bcm* eingetragen wird. Es ist darauf zu achten, dass dann das Schlüsselwort $EM nicht ebenfalls in diesem Forwardabschnitt steht!

30.5. System IDentifier (SID)

Systeme, die das WPROT Verfahren unterstützen, haben den Buchstaben "W" in ihrer SID. Durch diesen Buchstaben weiß die Mailbox, dass WPROT Daten anstelle von WP Daten unterstützt werden.

Hinweis: Ob E/M Daten der Diebox akzeptiert werden oder nicht, wird durch den Buchstaben "D" in der SID definiert.

Beispiel einer SID von OpenBCM:

```
[OpenBCM-1.04-AB1D1FHMRW$]
```

30.6. Syntax of WPROT mails

WPROT Daten werden an W@<Nachbar-Mailbox>, also z.B. an W@DB0ROF adressiert. Die Rubrik W wird benutzt, da es sich um eine einbuchstabige Rubrik handeln und somit für den allgemeinen Benutzer versteckt geführt wird (dies ist bei OpenBCM, DPBOX und auch DieBox so).

- Mailtitel: "WPROT Update" (sollte beim Empfang nicht überprüft werden)
- Typ der Mail: Usermail (Typ "P"(ersonal))
- Absender: das eigene Mailboxrufzeichen
- Empfänger: Nachbar-Mailbox
- Falls Lifetime unterstützt wird: 4 Tage (sollte beim Empfang nicht überprüft werden)
- Adressierung der Mail: W@<Nachbar-Mailbox>

Beim Mailempfang sollen nur Mails vom direkten Forwardpartner ausgewertet werden, die ein "W" in ihrer SID anbieten. Dies soll als Sicherheit dienen, da man nur bei direkten Forwardpartnern eine sichere Kommunikation voraussetzen kann.

Eine Nachricht kann willkürlich beliebig lang sein. Sie besteht aus individuellen Feldern, die jeweils eine Zeile lang sind. Die einzelnen Zeilen werden durch CR (wie in Packet Radio üblich) getrennt. Alle Zeichen sind 8bit groß. Es werden Standard ASCII-Zeichen genutzt.

(Hinweis: Nur 7bit Zeichen sind in ASCII definiert, alle Zeichen > 0x7f sind undefiniert).

Die Länge von einigen Feldern/Zeilen (z.B. Name, QTH) ist bei einigen Systemen limitiert. Falls ein Eintrag die lokalen Limits überschreitet, soll dieser Eintrag verworfen werden.

Beispielnachricht:

```
------
W < OE3DZW @OE1XAB  $F29OE3DZW005
------
WP Update
R:990407/0944z @:OE3DZW.#OE3.AUT.EU [BayCom-Mailbox] Bcm1.40P

D7 V 10
68 B OE3DZW.#OE3.AUT.EU 10 0 DL8MBT AX25 OE3DZW-8 923478289 1
```

```
5C M OE3DZW OE3DZW.#OE3.AUT.EU 749OE3DZW004 923471053 OE3DZW 1 "Dietmar" ? "?"
00 E WW OE3DZW F29OE3DZW007 F29OE3DZW006 OE3DZW 1 "#E"
```

30.7. Syntax of WPROT lines

Für jede Information wird genau eine Zeile genutzt. Eine Information hat folgendes globales Format:

```
    <checksum> <type> <data>
```

Die <checksum> ist eine einfache Byte-Checksumme, die bei 00 beginnt und über <flag> bis <data> gebildet wird. Sie beinhaltet keine Leerzeichen vor <flag> und kein CR nach <data>. Die Checksumme besteht aus einem zweistelligen Hex-Wert, der Grossbuchstaben verwendet (z.B. "4F").

Die <type> Deklaration kann mehrere bis maximal 10 Zeichen lang sein, normalerweise ist sie jedoch nur ein Zeichen lang.

Beispielzeile:

```
48 M OE3DZW OE3DZW.#OE3.AUT.EU F29OE3DZW004 919094319 OE3DZW 1 "Dietmar" ? "?"
^checksum
  ^type
    ^data (abhängig von <type>)
```

Generelle Regeln:

- Zeilen werden durch "<CR>" getrennt, andere Zeilenumbrüche werden nur beim Empfang akzeptiert.
- <type> werden durch Leerzeichen "0x20" getrennt; Tabulatoren oder andere White Space Zeichen sind nicht erlaubt (aber werden evtl. beim Empfang akzeptiert)
- Zeichen sind im allgemeinen in Grossbuchstaben zu benutzen, Kleinbuchstaben dürfen nur in bestimmten Fällen bei <data> angewendet werden (z.B. bei Namen, QTH)
- Falls ein Datenparameter Leerzeichen enthält/enthalten darf, müssen Anführungszeichen (z.B. "van Dark") benutzt werden.
- Falls ein Datenparamter unbekannt ist, soll ein "?" benutzt werden. Parameter dürfen nicht weggelassen werden.
- Zeitstempel sind generell im ANSI-Format (signed 32bit Integer) zu führen, 0 ist als der 1.1.1970 00:00 GMT definiert
- Alle Felder eines Eintrages sind zwingend auszufüllen

30.8. Line types

Bislang sind folgende Typen <type> definiert:

- V Version
- B BBS-Info
- E Remote-Erase
- M MyBBS-Info (Heimatmailbox)
- R Routing-Info

Die Typen V und M müssen mindestens von jedem System, das WPROT nutzt, unterstützt werden. Die anderen Typen dürfen beim Empfang ignoriert werden, besser ist natürlich, wenn diese von alle WPROT-fähigen Mailboxsystemen verstanden werden.

30.8.1. Type V

Dieser Typ wird genutzt, um die Version des unterstützten WPROT-Protokolls mitzuteilen.

Format:

<version>

<version> ist eine 32-Bit Integer-Zahl vom Typ "unsigned". Eine Zahl von z.B. "10" steht für Version 1.0. Eine Zeile vom Typ V sollte immer am Anfang einer WPROT Nachricht stehen.

Falls die empfangene Versionsnummer höher als die höchste bekannte Versionsnummer vom lokalen eigenen System ist, soll die WPROT Nachricht nicht ausgewertet werden.

30.8.2. Type B

Dieser Typ wird genutzt, um Mailbox-Informationen auszutauschen.

Format:

<h-Adr> <sysop call> <com prot> <hw-adr> <timestamp> <hop>

- <h-Adr> Hierarchische Adresse der Mailbox
- <sysop call> Rufzeichen des Sysops (ohne Name/MYBBS)
- <com prot> Protokolltyp, der für die Funkkommunikation verwendet wird, bislang sind folgende Typen definiert:
  - AX25.... Amateur-Radio Wireless Protokoll
  - TELNET.. RAW Protokoll
  - TCP/IP TCP/IP V4 Verbindung
  - ?..... Unbekanntes Protokoll
- <hw-adr> Abhängig vom Protokolltyp <com prot>:
  - AX25: AX25-Adresse mit/ohne via-Digis (z.B. "DB0AAB-8")
  - TELNET: IP-Nr. und Port, z.B. "db0aab.ampr.org:4719"
- <timestamp> ANSI-Zeitstempel vom Zeitpunkt, als die Nachricht erzeugt wurde
- <hops> Anzahl an Forward-Hops. Beim ersten Forward wird "hops" auf "1" gesetzt.

30.8.3. Type M

Dieser Typ wird genutzt, um Heimat-BBS-Informationen auszutauschen.

Format:

<call> <home bbs> <bid> <timestamp> <origin bbs> <hops> <name> <zip> <qth>

- <call> Rufzeichen des Users, dessen Heimatmailbox gemeldet wird
- <home bbs> Volle hierarchische Adresse der Heimatmailbox. Darf keine SSIDs beinhalten! Der hierarchische Teil der Adresse kann u. U. weggelassen werden, der linkste Teil der Adresse muss ein Mailboxrufzeichen sein. Beispiele:
  - OE3XSR.#OE3.AUT.EU -> gültig
  - OE3XSR-8.#OE3.AUT.EU -> ungültig
  - OE3XSR-9 -> ungültig
  - BBS.OE3XSR.#OE3.AUT.EU-> ungültig
  - OE3XSR -> gültig
- <bid> Es erscheint erstmal dumm, hier eine BID zu nutzen, aber: DieBox benutzt eine BID für jede MYBBS-Information, somit kann diese Information an dieser Stelle weitergegeben werden.

Falls die BID unbekannt oder nicht verfügbar ist, muss ein "?" an dieser Stelle genutzt werden. Hinweis: BIDs werden bei einigen Systemen nach Groß-/Kleinschrift unterschieden, deshalb darf an dieser Stelle die Groß-/Kleinschrift der BID niemals verändert werden!

- <timestamp> Zeitstempel der Heimatmailbox-Information
- <origin bbs> Rufzeichen der Mailbox, von der die ursprüngliche MYBBS Information herrührt, ohne Hierarchische Adresse und ohne SSID.
- <hops> Hops Zähler: Anzahl an Forwardboxen. 1 beim ersten Forward.
- <name> Name des Users, z.B. "Dietmar" oder "Hans-Jürgen"
- <zip> ZIP-Code (=Postleitzahl) des QTHs, z.B. "CH-3861"
- <qth> Heimat-QTH des Users, z.B. "Düsseldorf"

Beim Empfang macht es keinen Sinn, die BID in die lokale BID-Datenbank aufzunehmen!

30.8.4. Type E

Dieser Typ wird genutzt, um Remote-Erase(=Fernlösch)-Informationen auszutauschen.

Format:

```
<flood> <bbs> <bid> <erase_bid> <call> <hops> <comment>
```

- <flood> Verteiler der Nachricht, die gelöscht werden soll, darf länger als 6 Zeichen sein!
- <bbs> Rufzeichen der Mailbox, wo die Erase-Mail generiert wurde
- <bid> Es erscheint erstmal dumm, hier eine BID zu nutzen, aber: DieBox benutzt eine BID für jede MYBBS-Information, somit kann diese Information an dieser Stelle weitergegeben werden. Falls die BID unbekannt oder nicht verfügbar ist, muss ein "?" an dieser Stelle genutzt werden. Hinweis: BIDs werden bei einigen Systemen nach Groß-/Kleinschrift unterschieden, deshalb darf an dieser Stelle die Groß-/Kleinschrift der BID niemals verändert werden!
- <erase_bid> BID der Nachricht, die gelöscht werden soll
- <hops> Anzahl an Hops
- <comment> Grund, wieso die Nachricht gelöscht werden soll, z.B. "#K" oder "?"

Beispiel:

Die originale Nachricht:

```
-----------
TEST < OE3DZW @WW                       $F29OE3DZW00
-----------
R:990215/1734z @:OE3DZW.#OE3.AUT.EU [BayCom-Mailbox] Bcm1.40mg
From: OE3DZW @ OE3DZW.#OE3.AUT.EU
To:   TEST @ WW
TEST < OE3DZW @WW $F29OE3DZW006

(..)
```

wird durch

```
2F E WW OE3DZW F29OE3DZW007 F29OE3DZW006 OE3DZW 1 "#E"
```

gelöscht.

30.8.5. Type R

Dieser Typ wird genutzt, um Routing-Informationen auszutauschen.

Format:

```
<call> <version> <timestamp> <hops> <qual>
```

- <call> Rufzeichen
- <version> Version
- <timestamp> Zeitstempel der MYBBS
- <hops> Anzahl an Hops
- <qual> Qualität der Forwardgeschwindigkeit (Sekunden für das Versenden von 100 kBytes Daten, als ASCII Wert für einen Zahlenwert vom Typ "unsigned long")

Beispiel:

```
F3 R DB0SIF 10 950307915 2 16404
```

30.9. Calculation of routing quality for type R ("Active Routing")

Die Berechnung der Qualität einer Forwardroute kann durch Betrachtung der gesendeten Daten an eine Nachbarbox erfolgen, denn nur dadurch kann man verlässliche Informationen darüber gewinnen, ohne das spezielle Routingdaten ausgetauscht werden müssen.

Die einzige Möglichkeit, eine Zeit zu messen, ohne das ein Idle-Zustand das Ergebnis verfälscht, ist die Zeitdauer zwischen dem Startzeitpunkt, wo eine Mail gesendet wird (nicht der Start des Proposalvorschlags!) und der Empfangsbestätigung. Diese Zeitdauer kann dann auf einen Wert in Sekunden pro 100 kBytes umgerechnet werden.

Um die erzielte Bandbreite eines komprimierten FBB-Forwards widerzuspiegeln, wird nicht die Anzahl an gesendeten Daten zur Berechnung herangezogen, sondern das tatsächliche unkomprimierte Datenvolumen. Dadurch erhalten komprimierte Forward-Verbindungen einen besseren Qualitätswert als unkomprimierte bei gleicher Hardware-Bandbreite. Dies ist auch so beabsichtigt.

Kleine Datenmengen von weniger als 512 Bytes werden nicht zur Berechnung herangezogen, da hier der Einfluss anderer Faktoren eine vernünftige Berechnung beeinflusst. Wenn die letzte Messung mehr als 60 Minuten her ist, werden jedoch auch solche kleinen Datenmengen zur Berechnung herangezogen, bis der nächste, größer als 512 Bytes große, Datenblock gesendet wurde.

Die Qualität einer Forwardverbindung ist der Mittelwert aller Messungen in den letzten 5 Stunden. Dies verhindert allzu starke Sprünge in den Qualitätswerten.

Der letzte Mittelwert wird beibehalten, auch wenn in den letzten 5 Stunden keine neue Forwardverbindung mit der Partnermailbox stattgefunden hat.

Der Wert altert um jeweils 20% seines Anfangswertes pro Stunde aber mindestens um +1 pro Stunde. Nach einem Tag wird der Wert auf den Defaultwert einer unbekannten Forwardverbindung gesetzt (= 32767). Nach drei Tagen wird der Qualitätswert komplett aus der Forwardtabelle gestrichen.

Der maximale Qualitätswert ist 1 (d.h. 1 Sekunden für das Senden von 100 kBytes Daten), der minimale Qualitätswert ist 32767 (d.h. 32767 Sekunden für das Senden von 100 kBytes Daten).

Falls eine Forwardverbindung eine nicht-interaktive Methode verwendet (z.B. ein Dateiforward), wird die Qualität fest auf 16383 gesetzt. Eine Berechnung ist für solche Forwardverbindungen nicht möglich.

Bei jeder Erhöhung des Hop-Zählers wird der Qualitätswert jeweils fix um den Wert 100 plus 10% des ursprünglichen Qualitätswertes erhöht.

Später könnte auch eine Erhöhung des Qualitätwertes um die Anzahl an Mails, die noch in der Forwardschlange zu der Partnermailbox stehen, eingeführt werden. Dies ist aber im Moment nicht vorgesehen.

30.10. Routing of WPROT mails

Einige Einträge (Nachrichten mit Hop Zähler) dürfen zu anderen Mailboxen weitergeleitet werden, falls

- der Zeitstempel des Eintrags neuer ist, als der zuletzt empfangene (bei M und B Informationen) (*)
- die Erase-BID neuer ist (E Informationen)
- die Nachricht nicht älter als 1 Monat ist
- Die Nachricht soll nicht an den Absender zurückgeschickt werden
- Der Hop Zähler soll jedes Mal um eins erhöht werden
- Die Information soll nicht mehr weitergeleitet werden, wenn der Hop Zähler einen Wert von 50 erreicht hat.

(*) falls ein vorheriger Eintrag mit gleichen Daten bereits empfangen wurde, soll ein weiteres Update in einem speziellen Zeitfenster unterdrückt werden (z.B. B Informationen werden nur geupdatet, wenn sie sich verändert haben, oder einmal pro Monat)

Falls die lokale Mailbox einen speziellen Typ nicht unterstützt, werden Nachrichten ggf. nicht geforwardet. Die Typen E und B müssen nicht unbedingt unterstützt werden, M muss hingegen immer unterstützt werden.

30.11. Mixed WP/E&M/WPROT systems

Bei gemischten WP/E&M/WPROT-Systemen ist folgender Datenaustausch möglich:

```
WP   <-> WPROT (nur Typ M)
E/M  <-> WPROT (Typen M und E)
WP   <-> E/M   (nur Typ M)
```

Bei der Konvertierung von WP nach WPROT oder E/M wird der Zeitstempel vom Typ YYMMDD (z.B.: 990215) ins ANSI-Zeitformat gewandelt:
Sec: 00 Min: 00 Hours: 00 (GMT), -1 day (= -86400s) (!!)

Bei der Konvertierung von WPROT oder E/M nach WP wird der Zeitstempel durch Abschneiden der Stunden/Minuten gewandelt.

30.12. Security

Nur Nachrichten, die direkt vom Forwardpartner empfangen wurden, werden ausgewertet.

31. Forward specifications

The forward protocol was never specified anywhere. On the basics of W0RLI's mailbox [7] other authors developed a lot of variants. OpenBCM is using the additions of DF3AV, F6FBB and DL8HBS [2].

OpenBCM is compatible to minimum following systems: W0RLI, DieBox/TheBox, FBB, DPBox, MSYS, a lot of TNC based mini mailboxes. You can find a detailed description of the FBB and DieBox forward protocol in [11].

31.1. System IDentifier (SID)

Nach dem Verbindungsaufbau beim Forward tauschen die beiden Mailboxen Zeichen aus, welche als SID (**S**ystem **ID**entifier) bezeichnet werden. Dieser SID dient zum Aushandeln des Forwardprotokolls. Das beim anschließenden Forward verwendete Protokoll wird durch die Menge der von beiden Forward-Partnern unterstützen Features bestimmt.

Minimalkonsens:

- Der System-Identifier ist folgendermaßen strukturiert: "[f1-f2-f3]"
- Die Bindestriche (-) kennzeichnen das Ende des ersten Felds und den Anfang des letzten Felds.
- f1, f2 und f3 dürfen die Zeichen "[" oder "]" nicht enthalten.
- f1 ist die Identifizierung für den Autor der Software.
- f2 enthält softwarespezifische Daten. Es kann enthalten, was sich der Autor wünscht, zum Beispiel die Versionsnummer der Software. Dieses Feld darf Bindestriche (-) enthalten.
- f3 ist das Set von unterstützen Features. Es darf keine Bindestriche enthalten. Es enthält einen String von nicht-numerischen Zeichen, jeweils ein Zeichen für ein von der Box unterstütztem Feature. Jedes Zeichen kann auch von Ziffern gefolgt werden, diese geben die Revisionsnummer des Features an. Das Fehlen der Ziffer(n) ist mit der Revisionsnummer 0 gleichzusetzen.

Die OpenBCM-Mailbox sendet in der Regel folgenden SID:

```
    [OpenBCM-1.05-AB1D1FHMRW$]
                ^Unterstützte Features
            ^Versionsnummer
     ^Name der Software
```

Folgende Features sind in OpenBCM bekannt:

- A: Ack-Mails werden gesendet und verarbeitet
- B: Komprimierter Forward nach F6FBB
- B1: Resume-Mode beim Forward nach F6FBB
- D: DieBox-Erweiterungen
- D1: Block-CRCs beim Forward nach F6FBB
- F: Forward mit Fünferwechsel nach F6FBB
- H: Hierarchische Adressierung
- M: MIDs werden unterstützt (BIDs bei persönlichen Mails)
- R: Unterstützung erweiterter Reject-Meldungen
- W: Unterstützung von WPROT nach OE3DZW
- Z: Unterstützung des "CB-BCMNET Loginkonzeptes" (CB-Funk)
- $: BIDs (Bulletin-IDs) werden unterstützt

Das Feature D ist wie folgt definiert:

- Achtstellige Rubriknamen (Empfangsadressen bei Bulletins) sind erlaubt.
- Lifetimes werden im Box-Forward weitergeleitet.
- Forwarding von binären Mails nach dem AutoBIN-Protokoll.
- Forwarding von Erase- und MyBBS-Mails in einem besonderen Format.

Der Buchstabe D wurde von DL8HBS vorgeschlagen, er bedeutet so viel wie "D"L8HBS oder "D"ieBox. Die D-Features werden derzeit von folgenden Mailboxen verwendet:

- BayCom/OpenBCM-Mailbox (DL8MBT, OE3DZW, DG9MHZ, DH3MB, DK2UI, DH8YMB)

- Digi-Point-Mailbox (DL8HBS)
- DieBox (DF3AV, DL1BDY) - jedoch derzeit teilweise ohne "D" im SID
- WinGT-Mailbox (DG8NDL)
- MCuT-Mailbox (DG6VJ, DG4IAD)

Die Existenz des SIDs impliziert, dass das System die Richtung des Forwards ändern kann und OK/NO-Meldungen erzeugt.

Die OpenBCM-Mailbox wertet jedes empfangene SID aus. Bei der OpenBCM-Mailbox wird untersucht, ob das $-Zeichen in der SID enthalten ist. Falls nicht, erfolgt ein Disconnect, weil Systeme ohne BID nicht unterstützt werden.

Im SID anderer Boxen kann es auch folgende Zeichen geben:

- A (F6FBB): ACK-Messages sind bekannt (Forward-Befehl SA)
- B (F6FBB): nur in Verbindung mit F: Huffman-komprimierter S&F
- B1 (F6FBB): wie B, aber mit CRC am Beginn der Nachricht, zusätzlich Resume-Modus
- C (CBBS): Automatische Systemuhreinstellung (obsolet)
- D (DL8HBS): Rubriknamen achtstellig, CRC-16, in Verbindung mit B auch gepackt, CRC verbessert (in Verbindung mit B1)
- D1 (DL8HBS): wie D, zusätzlich Block-CRC-Mode im komprimierten Forward
- F (F6FBB): Fünferwechsel (Batch forward), F6FBB-Syntax verwenden
- H (W0RLI): System unterstützt hierarchische Adressen.
- I (W0RLI): System unterstützt ein "Null-Kommando", eine Zeile welche mit einem Strichpunkt (;) beginnt.
- L (G1NNA): Komprimierung (obsolet)
- M (W0RLI): Unterstützt MIDs (BIDs bei Usermails)
- R (AA4RE): Unterstützt erweiterte Reject-Meldungen
- X (W0RLI): X-Forward (Compressed batch forwarding)
- Y (WA7MBL): Forward mit dem binären YAPP-Protokoll
- $ (WA7MBL): Unterstützt Bulletin IDentifier; das Dollar-Zeichen muss der letzte Buchstabe im Feld f3 sein!

Einige Beispiele von SIDs:

- [RLI-17.5-HIX$]
  w0rli v17.5, unterstützt BID, H-Adr., Null-Kommandos und den X-Forward
- [CBBS-5.1-$]
  ag3f Version der rli/gyq cbbs.
- [FBB-7.00-AB1FHMX$]
  FBB v7.00, unterstützt ACK, Huffman-compressed Fwd mit Resume, X-Forward, H-Adr., MID, FBB
- [MSYS-1.16-H$]
  wa8bxn v1.16, unterstützt BID und H-Adr.
- [MBL-5.14-H$]
  wa7mbl v5.12, unterstützt BID und H-Adr.
- [4RE-2.3-MH$]
  aa4re v2.3, unterstützt MID, BID, und H-Adr.
- [THEBOX-1.9c1-DHM$]
  Diebox v1.9c1, unterstützt MID, H-Adr. und 8stellige Rubriknamen
- [DP-4.10-AB1DFHMR$]
  DigiPoint v4.10, F6FBB & DieBox Optionen werden unterstützt.

31.1.1. Rules for link setup with SID

Folgende Regeln gelten beim Austausch der SID zwischen zwei Mailboxen:
- Sende den SID in der ersten Zeile nach dem Connect. Beantworte den SID, wenn er empfangen wurde, mit einem kurzen Kommandoprompt ">".

- Wenn ein SID beim Connect empfangen wurde, antworte mit dem SID der eigenen Box.
- Das verwendete Protokoll ergibt sich aus der Schnittmenge der SIDs der beiden Forward-Partner.

31.2. RLI/Diebox forward

31.2.1. SEND command

Mit diesem Befehl wird eine Nachricht gesendet.

Sx TO [@ BBS[.LOC]] [< FROM] [$BID/MID]

x kann A, B, T oder P sein. Wenn x fehlt, so wird P angenommen, falls TO ein Rufzeichen ist, ansonsten wird ein B angenommen. Das Dollarzeichen $ ist kein Teil der MID, es kennzeichnet das Feld. Zwischen dem "$"-Zeichen und der BID darf kein Leerzeichen sein. Die Leerzeichen rund um den Klammeraffen @ können weggelassen werden. Beim User S&F muss das FROM-Feld mit dem Rufzeichen des Forwardpartners übereinstimmen. ST-Mails werden nicht angenommen.

31.2.2. OK/NO/REJ message

Die andere Box wird als Antwort auf das SEND-Kommando eine OK/NO/REJ-Meldung erzeugen, welche möglicherweise von irgendeinem Text gefolgt wird. Wenn die Antwort NO oder REJ ist, so wird sie von einem Prompt gefolgt. Wenn die Antwort OK ist, so muss der Forward-Partner (also jener, der SEND gesendet hat) die Mail forwarden. Normalerweise wird NO nur gesendet, wenn versucht wurde, eine Mail weiterzuleiten, deren BID bereits dem Empfangssystem bekannt war. REJ wird dann gesendet, wenn die Empfangsbox wegen eines Routingfehlers die Mail nicht annimmt. Eine solche Mail wird dann in die Datei *trace/sfhold.bcm* eingetragen. Eine Abfrage ist mit dem Befehl SFHOLD möglich.

Von den Meldungen "NO", "REJ" und "OK" muss jeweils nur der erste Buchstabe ("N", "R", "O") angegeben werden.

Zur Kompatibilität mit W0RLI/DPBox werden noch folgende Antworten unterstützt:

- LATER (W0RLI): Wie NO implementiert (Bedeutung: Empfange Mail bereits)
- HOLD (FBB): Wie OK (Nachricht wird Sysop vorgelegt)
- ERROR (FBB): Wie NO (Fehler im SEND-Befehl)

31.2.3. Transmitting a mail

Wenn mit OK geantwortet wurde, sendet die Nachbarbox die Nachricht. Die erste Zeile der Nachricht ist der Nachrichten-Titel. Die weiteren Zeilen sind der Text der Nachricht. Die Nachricht wird durch <CR><Ctrl-Z><CR> beendet, von der empfangenden Mailbox wird durch ein Linefeed und das Prompt (">") der Empfang bestätigt. Darauf wird von der sendenden Box eine weitere Nachricht angeboten.

31.2.4. Direction change

Falls keine Nachrichten mehr zum Forward bereitliegen, wird "F>" statt ">" als Prompt gesendet. Damit tauschen sendende Box und empfangende Box ihre Rolle.

31.2.5. End of forward connection

Wenn auch diese Box keine Nachricht mehr anzubieten hat, beendet sie das Forward durch das Senden der Zeichenfolge "***done".

31.3. FBB Forward specification

This chapter contains the original specification for the FBB forward from F6FBB.

The FBB forwarding protocols can operate either in ASCII or binary compressed modes.

There are three versions of this protocol. Each version is backwards compatible with the previous one. These versions are:
- ASCII basic protocol
- Binary compressed protocol version 0
- Binary compressed protocol version 1

31.3.1. ASCII basic protocol

The ASCII protocol in FBB software implements two forward protocols. The first one is the standard MBL/RLI protocol. The second one was developed for greater efficiency, particularly on long links where the command propagation delays occupy a significant portion of time. The exchange of commands and data is reduced to a minimum by sending several requests at a time. In normal VHF use up to five requests or messages are sent in one block. The data transfer direction is reversed after every block of data. This minimises the delaying effect of long links through Nodes and digipeaters, and also saves some time over short links (eg HF...).

FBB protocol is very simple in operation. It is based on MID/BID message identification. The protocol availability is indicated by the F letter in the SID (system type identifier contained in square brackets). All protocol command lines start in first column with the 'F' character. All protocol command lines are terminated by a return (CR) character.

This is the specification of the basic ASCII protocol. When I connect to another BBS that is FBB protocol capable, I will receive the SID followed by some text and the prompt (">"). The SID must contain the F flag. I send immediately my SID and the first proposal.

Proposals look like:

```
FB P F6FBB FC1GHV FC1MVP 24657_F6FBB 1345
F>
```

This means:

FB	Identifies the type of the command (proposal)
P	Type of message (P = Private, B = Bulletin).
F6FBB	Sender (from field).
FC1GHV	BBS of recipient (@field).
FC1MVP	Recipient (to field).
24657_F6FBB	BID ou MID.
1345	Size of message in bytes.
F>	End of proposal.

ALL the fields are necessary. This kind of command must hold seven fields. If a field is missing upon receipt, an error message will be sent immediately followed by a disconnection.

Messages are sent in blocks. There can be up to five FB command proposals per block. The number of command proposals is determined by the maximum size of a block of messages. In FBB software there is a parameter in INIT.SRV file which defines the maximum size of the message block. It is set by default to 10 kB for VHF use. It can be adjusted according to the quality of the link.

Example of proposal:

```
FB P F6FBB FC1GHV.FFPC.FRA.EU FC1MVP 24657_F6FBB 1345
FB P FC1CDC F6ABJ F6AXV 24643_F6FBB 5346
FB B F6FBB FRA FBB 22_456_F6FBB 8548
F>
```

This proposal is limited to three FB lines, as the total size of the messages overran the 10KB limit defined for this link.

When receiving the proposal, the other BBS will reject, accept or defer each message. This done with an FS line:

```
FS -+=
```

This means:
I don't want the first message (-). I need the second message (+). Defer the third message, as I'm already receiving it (=).

You would defer a message if you are already receiving it on another channel, or if you think that the message is too big, or for some other reason. The message should be proposed again at the next connection.

FS line MUST have as many +,-,= signs as lines in the proposal.

After receiving the FS lines, the block of messages will be sent. Each message is has:
- the title on the first line,
- the text,
- the Ctrl+Z in the last line.

Then the next message
- the title on the first line,
- the text,
- the Ctrl+Z in the last line.

And so on, until the number of messages in the block has been sent.

When the other BBS has received all the messages in a block, it implicitly acknowledges by sending its proposal for messages that it wants to send back to you, and thus the direction of transfer is reversed.

If there are no messages to send, it only sends a line:

```
FF
```

This line must not to be followed by an F>.
If the other side has no message (after receiving an FF), it sends a line:

```
FQ
```

and disconnects.

Example:

```
Connects xxxxx
Connected to xxxxx

[FBB-5.11-FHM$]
Welcome, Jean-Paul.
>

[FBB-5.11-FHM$] (F6FBB has the F flag in the SID)
FB P F6FBB FC1GHV.FFPC.FRA.EU FC1MVP 24657_F6FBB 1345
FB P FC1CDC F6ABJ F6AXV 24643_F6FBB 5346
FB B F6FBB FRA FBB 22_456_F6FBB 8548
F>

FS +-+ (accept the 1st and the 3rd)
Title 1st message
Text 1st message ......
```

```
^Z
Title 3rd message
Text 3rd message ......
^Z

FB P FC1GHV F6FBB F6FBB 2734_FC1GHV 234
FB B FC1GHV F6FBB FC1CDC 2745_FC1GHV 3524
F>

FS -- (Don't need them, and I send immediately the proposal)
FB P FC1CDC F6ABJ F6AXV 24754_F6FBB 345
F>

FS + (Accepts the message)
Title message
Text message ......
^Z

FF (no more message)
FB B F6FBB TEST FRA 24654_F6FBB 145
F>

FS + (Accepts the message)
Title message
Text message ......
^Z

FF (still no message)

FQ (No more message)

Disconnection
Disconnection of the link.
```

In this example, FBB protocol is used as the two BBSs had the F flag in their SIDs. If F6FBB had sent the SID [FBB-5.10-MH$] when answering FC1GHV, the protocol would have been standard MBL/RLI.

31.3.2. Binary compressed forward version 0

The compressed version of the protocol is an extension to the basic ASCII protocol. Compressed forward is indicated a letter B in the SID [FBB-5.15-BFHM$]. As it is an extension of the basic protocol, the SID must also have a letter F. A SID with just a letter B (and no F) will be treated as having neither letter.

In the message proposal section there are now two possible commands: FA means that the transfer will be an ASCII compressed message and FB means that the message will be a binary compressed file.

The submission of an ASCII message will be in the form:

```
FA P FC1CDC F6ABJ F6AXV 24754_F6FBB 345
```

The submission of a binary file will be in the form:

```
FB P FC1CDC F6ABJ F6AXV 24754_F6FBB 345
```

The actual message content is transfered in a different format from the ASCII protocol. The transfer is done in binary mode. The format used is derived from the YAPP protocol which is very reliable. Each message is made up of a header, blocks of data, an end of message marker and a checksum. This is directly equivalent to the transfer of one message in the ASCII protocol. Unlike YAPP transfers, there is no individual packet acknowledgement during the transmission of messages. The protocol is thus simpler and more efficient.

Format of header for an ASCII compressed message (type FA):

```
<SOH> 1 byte = 01 hex
Length of the header 1 byte = Length of the title and offset, including the two
  separating <NUL> characters
Title of the message 1 to 80 ascii bytes
<NUL> 1 byte = 00 hex
```

Offset 1 to 6 ascii bytes
<NUL> 1 byte = 00 hex

Format of header for a binary compressed file (type FB):

<SOH> 1 byte = 01 hex
Length of the header 1 byte = Length of the filename and offset, including the two <NUL> characters.
Name of the file 1 to 80 ascii bytes
<NUL> 1 byte = 00 hex
Offset 1 to 6 ascii bytes
<NUL> 1 byte = 00 hex

French regulations require that the title of the message or the filename are transmitted in readable ASCII and are not compressed.

The offset is also transmitted in ASCII and specifies the offset at which the data should be inserted in the file (in case of a fragmented file). In the version 5.12, this parameter is not utilized and is always equal to zero.

A data block contains from one to 256 bytes. It begins by two bytes which specify the format.

Data block format:

<STX> 1 byte = 02 hex
Size of data 1 byte = 00 to ff hex. if length is 256 bytes, the value is 00.
Data bytes 1 to 256 bytes
The last data block is followed by the end of file specifier and the checksum.
End of file specifier format :
<EOT> 1 byte = 04 hex
Checksum 1 byte = 00 to ff hex

The checksum is equal to the sum of all the data bytes of the transmitted file, modulo 256 (8 bits) and then two's complemented.
The checking of the checksum is very simple: The sum of the data bytes from the file and the checksum received modulo 256 shall be equal to zero.
In case of a checksum error, the message or the file is ignored and the system issues a disconnect request after having sent the comment:

***** Erreur checksum**

ASCII values of the characters (1 byte) used in the protocol:

<NUL> = 00 hex
<SOH> = 01 hex
<STX> = 02 hex
<EOT> = 04 hex

Most of ideas for this binary transmission come from YAPP protocol. Thanks to WA7MBL.

31.3.3. Binary compressed forward version 1

This protocol, used for the transfer of compressed ASCII messages or binary files, is an extension to the existing version 0 protocol. This version is indicated by the presence of the letters B1 in the SID:

[FBB-5.15-B1FHLM$]

As in version 0, there must also be a letter F in the SID for this version to be used.

The differences with regard to the version 0 are:
A variable number of extra fields in each submit line including at least the seven fields of the previous version.
A new set of answers in an FS line:

- + or Y : Yes, message accepted
- - or N : No, message already received
- = or L : Later, already receiving this message
- H : Message is accepted but will be held
- R : Message is rejected

- E : There is an error in the line
- !offset or Aoffset : Yes, message accepted from (Offset)

Most of these answers do not need explanation or were already used in previous version. + and Y, - and N, = and L, ! and A are equivalent but are still available for compatibility.
Aoffset asks the remote BBS to start transfer from Offset.

For instance, YLA3350RH (or +L!3350RH) means that:
1st message is accepted,
2nd message is delayed,
3rd message will be sent from offset 3350 (in compressed file),
4th message is refused,
5th message is accepted but will be held.

The submission of an ascii message will be in the form:

FA P FC1CDC F6ABJ F6AXV 24754_F6FBB 345

The submission of a binary file will be in the form:

FB P FC1CDC F6ABJ F6AXV 24754_F6FBB 345

The actual message content is transfered in a different format from the ASCII protocol. The transfer is done in binary mode. The format used is derived from the YAPP protocol which is very reliable. Each message is made up of a header, blocks of data, an end of message marker and a checksum. This is directly equivalent to the transfer of one message in the ASCII protocol. Unlike YAPP transfers, there is no individual packet acknowledgement during the transmission of messages. The protocol is thus simpler and more efficient.

Format of header for an ASCII compressed message (type FA):

<SOH> 1 byte = 01 hex
Length of the header 1 byte = Length of the title and offset, including the two separating <NUL> characters
Title of the message 1 to 80 ascii bytes
<NUL> 1 byte = 00 hex
Offset 1 to 6 ascii bytes
<NUL> 1 byte = 00 hex

Format of header for a binary compressed file (type FB):

<SOH> 1 byte = 01 hex
Length of the header 1 byte = Length of the filename and offset, including the two <NUL> characters.
Name of the file 1 to 80 ascii bytes
<NUL> 1 byte = 00 hex
Offset 1 to 6 ascii bytes
<NUL> 1 byte = 00 hex

French regulations require that the title of the message or the file name are transmitted in readable ASCII and are not compressed.

The offset is also transmitted in ASCII and specifies the offset at which the data should be inserted in the file (in case of a fragmented file). In the version 5.12, this parameter is not utilized and is always equal to zero.

A data block contains from one to 256 bytes. It begins by two bytes which specify the format.

Data block format:

<STX> 1 byte = 02 hex

Size of data 1 byte = 00 to ff hex. If length is 256 bytes, the value is 00.
Data bytes 1 to 256 bytes

The first transmitted block of data must contain a header containing:
- the CRC16 of the full binary file (2 bytes)

- the size of the full uncompressed file (4 bytes)

This data is in little-endian Intel format (less significant first).

The last data block is followed by the end of file specifier and the checksum of the data sent.

End of file specifier format:
<EOT> 1 byte = 04 hex Checksum 1 byte = 00 to ff hex

The checksum is equal to the sum of all the data bytes of the transmitted data, modulo 256 (8 bits) and then two's complemented.

The checking of the checksum is very simple:
- The sum of the data bytes from the file and the checksum received modulo 256 shall be equal to zero.
- In case of a checksum error, the message or the file is ignored and the system issues a disconnect request after having sent the comment:
***** Erreur checksum**

A CRC16 is computed for the full binary file including the length of the uncompressed file (4 bytes in top of file). In the case of a resume, it will be the only means available to ensure that all the parts of the message or file has been received correctly.

The LZHUF_1 program, when used with option "e1", generates a binary compressed file in the following format:
CRC16 : 2bytes Length: 4 bytes Data : rest of the file

In case of forwarding with a BBS using version 0, only the part from offset 2 will be sent.

In case of forwarding with a BBS using version 1, the 6 top bytes will be always sent, then if resume seek to asked offset, then send data.

ASCII values of the characters (1 byte) used in the protocol:
<NUL> = 00 hex
<SOH> = 01 hex
<STX> = 02 hex
<EOT> = 04 hex
Comments will be welcome.

32. DFWD specification v0.1

In this chapter you will find a brainstorming from OE3DZW for a concept of DFWD. This is not implemented at the moment in OpenBCM! DFWD means "direct forward". Usermails should be send directly from source mailbox to each target mailbox without forwarding with other mailboxes.

32.1. Abstract

Direct forwarding (no relaying) between packet radio mailboxes as an alternative to the store & forward conception, that is used today.

32.2. Introduction

Today we have a very dense network of BBSes and digis. I takes seconds to leave a mail in a remote BBS, but can take hours..days to forward a personal mail from a local BBS to the same destination - if it arrives

at all. The mails is forwarded and stored in several BBSes, at each hop there is a chance of:
- Misconfiguration by the local sysop
- Usage of old or buggy software
- Inconsequent routing due to the usage of different routing concepts
- Unavailability of BBS
- Congestion of a BBS due to overload (eg. big bulletins, usermail has to wait until fwded)

In this specification direct forward between two BBS is defined.

32.3. How it is working

Which mails should be forwarded directly from A to B:
- Address: <valid_call> > <valid_call> @ <B>
- Mailtype P
- May originate in A (or not)

There is no technical reason why not to forward mails from or to no-calls (eg. servers). But servers are usually causing trouble, therefore leave them out now, they can be added later.

How to know that the remote B supports DFWD
- received WPROT-broadcast from that BBS

(or any similar method, eg. sysop-configuration)

That broadcast contains
- Flag that DFWD is accepted
- Connect Path to the remote BBS

How to connect to the remote BBS

A Flexnet-compatible network is assumed, it can be modelled as:
- call+active-ssid of BBS A
- uplink network (eg. local digi)
- fwd-call (ssid) of BBS B

Eg: BBS A knows that it needs to connect to OE1XLR to get access to the network (due to sysop configuration) and has a defined bbs-call and "fwdssid (ssid used for outgoing connects). BBS B has broadcasted its "fwd"-access eg. DB0AAB-7
Therefore the connect path is in Pseudo-Syntax:

```
    DB0AAB-7 v OE1XLR < OE1XAB-5
```

Note: Some software is using only a single SSID for both - user-logins and forwarding, some mix them. Here we are only talking about the SSID used for forwarding.
(FBB,DPBOX: same ssid for user-logins and forwarding, BCM: different ssid, but BBS-forwarding is for historical reasons also accepted on SSID which should only be used for user-logins; DFWD should never use the SSID used for user-logins on BCM).

How to know that BBS B is available on the network?

This is up to the implementation. The BBS might track the Flexnet-Routing information, but it could also just try to connect and see.

What should happen if the mail can not be forwarded?

-Timeout: try DFWD for eg. 6min, if that fails use "normal" forward

The exact timeout (or if DFWD is tried at all) is up to the implementation.

The current forwarding

is done as usual. If there is more than one mail in the queue, more than one mail is forwarded. If there is mail from B to A it is also forwarded. The only limitation is:
- personal
- to <call>@<local_bbs> (no relaying)
- from <call>

There is no special SID flag used.

Security?
Usually forward is secured by a password, but here only public key algorithms could be used for practical reasons.
But: There is no relaying, no bulletins are forwarded, therefore a "bad" user can not do more than he could do when logging into some BBS using any call.
Still: No WP-broadcasts should be accepted in DFWD, no logins from BBSs which are not known to support DFWD should be accepted.

If there is some password set (e.g. some sysops have aggreed on a "common" password), that password and password-method should be used.

33. DIDADIT specification v0.91

In this chapter you will find the specifications of DIDADIT protocol v0.91, as it was published at http://1409.linkt.de.

33.1. Assumptions

All binary numbers are little-endian (Intel style). All alphanumerical strings are ISO-8859-1 coded.

33.2. Basic concept

Data is transported block by block. Each block consists of

- a block delimiter (EOB),
- the data and
- a CRC.

The whole block (including data) is stuffed by means of the SLIP/KISS pattern (see chapter "33.9. Stuffing").
The CRC is mandatory and included in every block. Blocks with an incorrect CRC are discarded, as the reason for the CRC error cannot be determined.
The used CRC polynome is the same as used by the AutoBIN protocol:
<EOB><DATA><CRC><EOB>

33.3. Block types

The first two bytes of a block always specify the block type. The block type identifies the kind of following data:
<TYPE><DATA>

If a block requires an MD5 hash (noted in this spec with "**MD5: yes**"), it is included between the block type and the data:
<TYPE><MD5><DATA>

All block types except <DATA> require an answer block. No other non-DATA block may be sent until the awaited reaction has been received. If the reaction is not received for a certain timespan (implementation specific), the first block is repeated. After five retries the connection is aborted.
Certain block types contain an MD5 hash after the type tag. It is used to help a third, monitoring station identify the transferred file.

33.3.1 INFO block

This block contains a list of TAG=VALUE pairs. Each pair is followed by a <CR>.
Currently defined tags:

Tag	Value type	Description
FILENAME	string	Name of the file being transferred.
PATH	string	Directory in which the file is written on the receiving machine.
SIZE	decimal	Size of the file in bytes.
MD5	Hexadecimal ASCII	MD5 hash of the complete file.
TIME	decimal	TIME is decimal as epoch-time (seconds since 1970).
BLOCKSIZE	decimal	Size of a transfer block.
VERSION	string	DIDADIT version being used.
FEATURES	string	This field specifies DIDADIT extentions. The following strings are currently defined: CHAT

A TAG=VALUE-pair may be no longer than 512 bytes.
MD5: no
Reaction: START or ERR

33.3.2 START block

Supercedes "#EOK#" from the first draft.
Currently defined tags:

Tag	Value type	Description	Comment
OFFSET	decimal	as offset for a resume. If OFFSET=SIZE, the other station has to send a FIN-block immediately and then waits for a REQ-block.	
PARTMD5	hexadecimal	The file's MD5-hash from beginning to OFFSET. Used to check if we still are transferring the same	This command may only be sent if the receiver cannot determine whether the sent file is the same as the partially received one. In this case, the transmitting station *must* check whether its

		file.	file hash up to the specified OFFSET is the same as the here transmitted MD5-hash. If they are the same, the transfer starts at the specified offset. Otherwise the transfer is started at offset 0.
BLOCKSIZE	decimal	Set, if the value announced in INIT is too high.	
VERSION	string	DIDADIT version being used.	
FEATURES	string	see INFO block	

A TAG=VALUE-pair may be no longer than 512 bytes.
MD5: no
Reaction: DATA or FIN

33.3.3 ERR block

General negative response.
The block contains an error code followed by the reason in plain text, e.g. "100 Unexpected block type". If the second digit in the error code is smaller than 5, it is a fatal error. If it is 5 or larger, then it is non-fatal.
The complete string may not be longer than 80 bytes.
A list of error codes is supplied in chapter "33.7. Error codes".
MD5: yes
Reaction: disconnect, end of protocol or specific reaction according to error code

33.3.4 DATA block

A DATA block contains the following information:

- 4 bytes offset
- 2 bytes blocklength
- 0x0-0xFFFF bytes of data

<OFFSET><LEN><DATA>
The receiver should check whether the combination of offset and blocklength is valid. If not, it should send error 104.
MD5: yes
Reaction: not neccessary

33.3.5 REQ block

A REQ block requests one or many parts of the file. This can be done at any time during the transfer, but usually it will be applied after the full filelength has been sent. If the requested block is larger then the block size agreed upon, it has to split up into several DATA blocks.

- 4 bytes offset1
- 4 bytes length1

- 4 bytes offset2
- 4 bytes length2
- ...

A REQ block can contain more than one request. After a FIN block has reached the receiver, only one REQ block can be sent. If the sender has answered all requests, it transmits a FIN block again and waits for the answer. This procedure is repeated until the receiver has aquired the file completely.
MD5: yes
Reaction: DATA or ERR

33.3.6 FIN block

After the file has been sent completely, the transmitter sends a FIN block. This block can optionally contain data in a not yet defined format. After reception of FIN, the receiver checks for missing parts and, if neccessary, requests them with REQ block.
MD5: yes
Reaction: REQ or FIN-ACK

33.3.7 FIN-ACK block

As soon as the client has received all data, it closes the transfer by submitting a FIN-ACK.
MD5: yes
Reaction: end of protocol

33.3.8 ECHO-REQUEST block

Request echoing of the supplied data block. The response is sent in an ECHO-REPLY block.
MD5: no
Reaction: ECHO-REPLY

33.3.9 ECHO-REPLY block

Send data received by ECHO-REQUEST block back to the originating station.
MD5: no
Reaction: not neccessary

33.3.10 ABORT block

Can be sent by either station at any time. The DIDADIT connection is disengaged. The aborting station will not accept any further DIDADIT blocks.
MD5: no
Reaction: dependant on implementation (terminal mode, disconnect)

33.3.11 CHAT block

Used for chat between the station's operators. The text should be displayed immediately.

This block type may only be used if given in the INFO and START block.
MD5: no
Reaction: not neccesary

33.4. Starting a DIDADIT transfer

A DIDADIT transfer is initiated by the string

<cr>#DIDADIT#<cr>

directly followed by an INFO block containing file information.
The receiving station must either answer with

<cr>#OK#<cr>

followed by a START-block, or

<cr>#ABORT#<cr>

if the transfer shall be aborted. This can be sent manually, if the receiver is not capable of DIDADIT.

33.5. Implementation details

The following has been a recommendation in previous versions of this document but is mandatory now.

- Send data in original order. Simple implementations perhaps do not use an index file and, in case of an error, would produce too many requests for block retransmission.
- For the same reason the transmitter should quickly react upon requests from the receiving station, especially REQ blocks.

33.6. Compression

Compression is an optional protocol extension. It has been developed by Marco, DL8NJY and is implemented in WinSTOP.
If an implementation supports compression, it will place a supplementary line into its INFO block:

COMPRESS=Type1,Length1[, Type2,Length2]...

where TypeX is the compression type and LengthX is the compressed file size if using this compression. Currently supported types are LZH as implemented in FBB forwarding and gzip (standard GNU zip). If the receiving station supports one of the offered compression types it adds

COMPRESS=Type

to its START block. Then the sender starts transmission of the compressed data. Please notice that the filesize is now the value given in the COMPRESS line. The value given by the SIZE line regards the uncompressed file.

33.7. Error codes

Protocol errors:

Code	Text equivalent	Error description
100	Unexpected block type	The received block does not fit in the current situation (fatal).
101	REQ block with data out of range	A REQ block with an offset below 0 or beyond filesize has occured.

102	MD5 validation error	A block contained a wrong MD5 hash with is different from the currently transferred file.
103	PARTMD5 format error	Wrong count of characters.
104	Invalid offset/blocksize	The given data are not in the current file (e.g. offset > filesize).
150	Unknown block type	The received block does not fit in the current situation (non-fatal).

Errors within START/INFO blocks:

200	Illegal line in INFO block
201	Illegal line in START block
202	INFO block without MD5 info
203	Empty file or missing size info in INFO block
204	INFO block without BLOCKSIZE info
205	INFO block without FILENAME info
206	Block corrupt
208	Compression type not supported

Implementation dependent errors:

300	Could not create file

33.8. List of all block types

Number	Type
1	INFO
2	START
3	ERR
4	DATA
5	FIN (WARNING: this has changed recently)
6	REQ (WARNING: this has changed recently)
7	FIN-ACK
9	ECHO-REQUEST
10	ECHO-REPLY
11	ABORT
12	CHAT

33.9. Stuffing

Every block is stuffed using the following system:
TX:

- FEND := FESC+TFEND
- FESC := FESC+TFESC

RX:

- FESC+TFEND := FEND
- FESC+TFESC := FESC

Constants:

Name	Code	Name	Code
FEND	0xC0	FESC	0xDB
TFEND	0xDC	TFESC	0xDD

On TX, a FEND- or FESC-byte will change to the two-byte code; on RX, the given combinations are switched back to the FEND- or FESC-byte. After stuffing is done, the stuffed data will *nowhere* contain the FEND-byte. This is our EOB-code.

34. Further specifications

34.1. Lookout of a mail

Die Nachrichten selbst werden jeweils in einer eigenen Datei abgelegt. Der Dateiname entspricht den im Kapitel "24. Specification of system files" erwähnten Gepflogenheiten.

Die ersten vier Zeilen einer Nachricht haben organisatorische Bedeutung. Sie werden beim Auslesen der Nachricht grundsätzlich nicht ausgegeben, sondern ausgewertet.

Im Unterschied zu den rein maschinell erzeugten Dateien *list.bcm* und *check.bcm* können die Nachrichten selbst sehr wohl editiert werden. Auch die Form ist relativ tolerant.

Erste Zeile der Nachricht:

```
HUMOR < OE3DZW   DB0LNA @DL $04B4OE3DZW0E #30 %!#!!2 =!!!|B!!!!!!!!!!!!!!!
```

Im Prinzip haben die Felder in dieser Zeile exakt die gleiche Bedeutung wie in *list.bcm*. Allerdings wird hier nicht nach Feldlänge ausgewertet, sondern nach den entsprechenden Operatoren ("<", "@", "$", "%", "#", "=", "|") gesucht. Die Reihenfolge ist absolut unerheblich, ebenso die Feldlänge. Blanks dürfen an jeder Stelle eingefügt werden. Dadurch kann jede Nachricht problemlos per Hand editiert und umgebaut werden, es können sogar Nachrichten händisch erzeugt und gelöscht werden.

Die Anzahl der Zeilen und Bytes (nach dem "%"-Operator) hat rein informativen Charakter, sie wird nicht überprüft und anderweitig verwendet, sondern nur beim Befehl READ ausgegeben. Bei der Anzahl der Bytes wird bei AutoBIN-Mails die Länge des BIN-Anteils, bei Text-Mails nur die Nachricht selbst, also ohne R:-Zeilen angegeben. Dadurch ist die angezeigte Größe der Nachricht nicht von der Länge des Forward-Weges abhängig. Der Operator "=" kennzeichnet den Offset des AutoBIN-Teils der Mail, "=!!!" bedeutet, dass die Mail keinen AutoBin-Teil enthält. (Kodierung: "!" entspricht 0, jede Stelle läuft von 0-127 (7 Bit);

7+7+7=21 Bit entsprechen einem maximalen Zahlenwert von 2097151 ("unsigned int" in C), das ist also der maximale Offset). Mit dem Parameter "|" werden 16 Byte-Flags markiert.

Die 2. Zeile betrifft das Forwarding:

```
DB0WGS  OE1XAB  OE5XEM* .......
```

In dieser Zeile stehen Boxrufzeichen, zu denen die Mail geschickt werden soll bzw. bereits geschickt wurde. Ein * hinter jedem Rufzeichen bedeutet, dass die Mail bereits geschickt wurde.

Wichtig ist, dass die Zeile mit einer hinreichend großen Anzahl von Platzhaltern beschrieben ist, da auf die Zeile im Laufe des Forwardingbetriebes schreibende Zugriffe erfolgen, um die Rufzeichen zu markieren. Als Platzhalter sind entweder Blanks oder Punkte zulässig.

Ein "*" am Anfang der Forwardzeile markiert die Nachricht als gelöscht.

In der 3. Zeile wird vermerkt, wer die Nachricht bereits gelesen hat. Das Format ist ähnlich wie bei den Forwardcalls, es wird allerdings nur ein Blank zwischen den Rufzeichen freigelassen. Als Platzhalter werden auch hier Punkte verwendet. Diese Zeile hat eine Länge von 79 oder 252 Zeichen (je nach Version).

In der 4. Zeile steht der Betreff der Nachricht. Dieser darf im Prinzip beliebig lang sein, muss jedoch in einer Zeile stehen.

In der 5. Zeile steht "From:", gefolgt vom Absendercall und Call der Box, in der die Mail eingegeben wurde:

```
From: <absender-call> @ <boxadr>
```

Stimmt die Absendermailbox <boxadr> nicht mit der MyBBS des Absenders überein, d.h. wird die Mail nicht von der MyBBS aus gesendet, so wird eine zusätzliche Zeile eingefügt:

```
Reply-To: <absender-call> @ <absender-mybbs>
```

Beim Empfang einer Mail wird der Name in der From-Zeile ausgewertet und falls der Name des Absenders im *users.bcm* noch nicht bekannt ist, dort eingesetzt. Das Rufzeichen <absender-call> in dieser Zeile muss mit dem Absenderrufzeichen übereinstimmen, sonst wird der Name ignoriert. Es wird neben "From:" auch "de" (DieBox) ausgewertet.

In der nächsten Zeile steht "To:", gefolgt vom Empfängercall und der vollständigen Empfängeradresse.

Wird die Nachricht im User-S&F empfangen, so wird eine Zeile

```
X-Info: Received in User S&F from OE3DZW at OE3XSR.#OE3.AUT.EU
```

eingefügt. Wird die Mail von einem User eingespielt, der sich an der Einspielbox kein Login-Passwort gesetzt hat, so wird eine Zeile

```
X-Info: No login password bzw.  X-Info: No upload password
```

zusätzlich eingefügt. Die X-Info-Zeilen werden beim Empfang nicht ausgewertet.

Die nächste Zeile ist immer eine Leerzeile.

Hinweis: In früheren Versionen war die Länge des Headers fix, damit war die Leerzeile (und damit der Beginn des Inhalts der Mail) ebenfalls fix in der 8. Zeile.

In der nächsten Zeile beginnt der Inhalt der Nachricht.

Zusammengefasstes Beispiel:

```
      HUMOR < OE3DZW ~ DB0LNA @DL $04B4OE3DZW0E #30   %!#!!2 =!!!
      DB0WGS OE1XAB OE5XEM .....................................bis Pos. 79
      ..........................................................bis Pos. 252
      Langweiliger Titel
```

```
R:941104/0119z @:OE3DZW.#OE3.AUT.EU [BayCom-Mailbox] bcm1.39
From: OE3DZW @ OE3DZW.#OE3.AUT.EU (Dietmar)
To  : HUMOR @ DL
Reply-To: OE3DZW @ OE1XAB.#OE1.AUT.EU
X-Info: No login password

Eine kurze Mail <CR><LF>
```

Eine Mail mit einem AutoBIN-Teil hat ein ähnliches Aussehen:

```
HUMOR < OE3DZW @OE3XSR.#BAY.DEU.EU      $04B4OE3DZW0F #123 %!#!!, =!%}
DB0WGS ..................................................bis Pos. 79
.........................................................bis Pos. 252
Ein kurzweiliger Titel
R:941104/0120z @:OE3DZW.#OE3.AUT.EU [BayCom-Mailbox] BCM1.39
From: OE3DZW @ OE3DZW.#OE3.AUT.EU (Dietmar)
To  : HUMOR @ OE3XSR.#BAY.DEU.EU

Das ist der Text einer binären Mail.<CR><LF>
#BIN#10#|43301<CR><NUL><NUL>...<NUL>
```

Direkt anschließend folgt der binäre Teil der Nachricht

```
<NUL> = 0x00, <CR> = 0x0D, <LF> = 0x0A
```

Beim Empfang von Mails wird die Zeichenfolge "//e " am Zeilenanfang erkannt und der erste "/" gegen "\" ersetzt. Dies soll die Verbreitung von Mails unterbinden, die Terminals durch den "//ECHO"-Befehl dazu veranlassen, beliebige Befehle an die Mailbox zu schicken.

34.2. Mails with AutoBIN parts

In der OpenBCM-Mailbox ist ein zur DieBox kompatibles AutoBIN eingebaut.

Unter binären Mails werden Nachrichten verstanden, welche eine beliebige Zeichen in einer beliebigen Reihenfolge enthalten dürfen. Das Übertragungsprotokoll beruht darauf, dass die Dateilänge am Anfang der Übertragung angegeben wird. Die Datei selbst wird transparent übertragen.

34.2.1. Sending of AutoBIN mails

Der Befehl SEND hat die gleiche Syntax wie bisher. Wird jedoch ein "<CR>#BIN#" im Datenstrom erkannt, sucht die OpenBCM-Mailbox nach Zahlen von 0-9, die nach dem zweiten # folgen. Die damit dargestellte Anzahl von Zeichen wird eingelesen und sodann die Datei geschlossen.

Die generelle Zeichensequenz zur Einleitung eines AutoBIN-Transfers lautet:

```
<CR>#BIN#<länge>#|<CRC>#Dateiname<CR>
```

Verbindlich ist dabei allerdings nur das "#BIN#" sowie die Längenangabe. Der Dateiname wird nicht ausgewertet, sondern transparent durchgereicht.

Beispiel:

```
S OE3DZW BINTEST<CR>
Es folgt ein kurzer Binärteil<CR>
```

(Hier startet der Benutzer die AutoBIN-Übertragung bei seinem Terminalprogramm, der Rest geht automatisch.)

```
-> Terminal-Programm:
<CR>#BIN#10#|23412#$1234567#C:\BOX\TEXTE\BIN.EXE<CR>

-> Box: #OK#<CR> (nicht im S&F)

-> Terminal-Programm: 123456789<CR> (Das sind genau zehn Zeichen!)
```

Wichtig: Die "#BIN#"-Sequenz wird komplett gespeichert, die OpenBCM-Mailbox wertet aber nur die in der Sequenz übermittelte Anzahl von Zeichen und die CRC aus. Die Sequenz ist auf eine Zeile beschränkt und muss mit einem <CR> (HEX 0D) abgeschlossen werden. Das nächste Zeichen nach dem <CR> ist das erste Zeichen des Binärteils. Die Anzahl der Zeichen des Binärteils muss unmittelbar nach der "#BIN#"-Sequenz folgen und darf nur die Ziffern '0' bis '9' enthalten. Bei jedem anderen Zeichen wird die Auswertung der Längenangabe abgebrochen.

Die CRC wird nach einem von DC4OX in den Boxen veröffentlichten Verfahren berechnet. Wurde keine CRC mitgesendet, so wird von der OpenBCM-Mailbox diese selbständig berechnet und hinzugefügt. Falls die berechnete mit der übermittelten CRC nicht übereinstimmt, so wird eine Fehlermeldung ausgegeben und die Nachricht nicht abgespeichert.

Das 16 Bit-Prüfpolynom bietet nur einen recht beschränkten Schutz gegen Übertragungsfehler. Es können damit Einzelbitfehler nur bis zu einer Größe von 8 kB sicher erkannt werden. Bei größeren Dateien können Einzelbitfehler unerkannt bleiben, die erreichte Übertragungssicherheit ist jedoch immer noch höher als ohne jede Prüfsumme, zumal meist nicht Einzelbitfehler auftreten, sondern von Digis oft Teile von Paketen verschluckt werden - womit es zur "Fehlersicherung" ausreicht die Dateilänge zu vergleichen – und diesen Zweck erfüllt das 16 Bit-Polynom.

In den Mail-Dateien wird der Binär-Start in der Headerzeile nach dem Operator "=" im komprimierten Format angegeben. Am Binärstart ungleich "=!!!" (entspricht 0) erkennt man eine AutoBIN-Mail. Sie darf keinesfalls mit einem Editor bearbeitet werden, da sonst evtl. der Inhalt oder der Offset nicht mehr passt. Am Beginn des eigentlichen Binärinhalts wird die #BIN#...-Zeile gespeichert und auf insgesamt 80 Zeichen mit <NUL> (nicht blank) aufgefüllt.

Der Titel im *list.bcm* ist mit dem Zusatz "(BIN)" versehen. Dadurch unterscheidet sich der Titel im *list.bcm* von dem in der Nachricht selbst. Der Zusatz "(BIN)" wird auch nicht geforwardet.

Hier für die Berechnung der Prüfsumme ein Implementierungsbeispiel in 'C':

```
 /*
  *   CCITT Polynom x^16 + x^12 + x^5 + 1
  *
  *   hier im Gegensatz zu CCITT, (A)X.25:
  *     - Datenbyte MSB zuerst
  *     - CRC-Start mit 0
  *     - keine Multiplikation Nachricht mit x^16
  *       (CRC wird bei Empfang nicht über CRC-Bytes berechnet)
  *     - kein Invertieren des CRC
  *
  *   wird z.B. benutzt von:  UoSat DCE, SEVEN
  */
 unsigned crctab[256];
 unsigned crc;
 unsigned byte;        /* Datenbyte, high Byte = 0 ! */
 /*
  *   entweder beim Programmstart einmal aufrufen,
  *   oder die 512 Byte Tabelle einmal ausrechnen lassen und dann als
  *   initialisierte Tabelle ins Programm übernehmen
  */
 void init_crctab(void)
   { unsigned n;
     unsigned m;
     unsigned r;
     unsigned mask;
     static unsigned bitrmdrs[] = { 0x9188,0x48C4,0x2462,0x12*,
                                    0x8108,0x4084,0x2042,0x1021  };
     for (n = 0; n < 256; ++n)
       { for (mask = 0x0080, r = 0, m = 0; m < 8; ++m, mask >>= 1)
           if (n & mask) r = bitrmdrs[m] ^ r;
 /*
  *       "if (n & mask) r ^= bitrmdrs[m];"  ist kürzer, aber
  *       manche Compiler übersetzen das falsch
  */
```

```
        crctab[n] = r;
      }
    }
  /*
   *   byteweiser Algorithmus, C, portabel
   */
  void do_crc(void)
    {
      crc = crctab[crc >> 8] ^ (crc << 8 | byte);
    }
```

34.2.2. Reading of AutoBIN mails

Hier ist die meiste Arbeit durch die Terminalprogramme zu erledigen.

Von der OpenBCM-Mailbox aus ist das Auslesen einer Nachricht mit Binärteil kaum anders als das normale Auslesen einer Nachricht. Sie erkennt intern den binären Zustand und unterbindet damit jegliche Unterdrückung nicht erwünschter Zeichen. Das ist alles!

Wichtig für Programmierer von Terminal-Programmen: Die "<CR>#BIN#..."-Sequenz kann auch über Frame-Grenzen hinweg beim Terminalprogramm ankommen kann. Die Terminalprogramme sind verantwortlich für die Erkennung der Sequenz und haben entsprechende Vorbereitungen zu treffen, um den Binärteil auf die Festplatte zu schreiben.

Wichtig ist für den Binärteil nur das <CR>#BIN#<NUMMER><CR>. SP z.B. benutzt noch weitere Elemente, die auch mit ausgesendet werden, sie sind für die eigentliche Übertragung des Binärteils aus der Sicht der Mailbox jedoch ohne Belang.

Wichtig ist weiterhin, dass vor dem Binärteil einer Nachricht noch ein beliebiger anderer Text erscheinen kann. Die Terminalprogramme müssen also auf die #BIN#- Sequenz warten, und dürfen den Status nicht ändern, es sei denn, der Benutzer verlangt es.

Es kann immer nur ein Binärteil in einer Nachricht übermittelt werden.

Den Abschluss des Binärteils bildet die Zeichenfolge <CR>#OK#<Prüfsumme><CR> der von der Box ausgesendet wird (nicht im S&F).

Falls das Terminalprogramm "#OK#", "BIN-RX" oder "BIN-TX" aussendet, so wird dies von der Box ignoriert.

Die Option "-X" beim READ von AutoBIN-Mails bewirkt, dass nur der Textteil der Nachricht ausgegeben wird. Das Lesen und Schreiben einer AutoBIN-Mail kann nicht abgebrochen werden.

34.2.3. Forward of AutoBIN mails

Binäre Nachrichten werden genauso wie gewöhnliche Text-Nachrichten entsprechend der Eintragungen in *fwd.bcm* weitergeleitet. Beherrscht die Nachbarbox AutoBIN nicht, so wird eine Nachricht erzeugt, welche angibt, wo die AutoBIN-Mail liegt.

Aktiviert wird AutoBIN beim Forward durch eine der Bedingungen:

- Option "-I" in *fwd.bcm*
- SID-Kennung D

Ist eine der Forward-Partner der eigenen Box eine DieBox, so ist bei dieser DieBox in der Datei *mbsys\sfwid.box* folgender Eintrag zu machen:

```
BayCom-1.1  18 S
```

```
BayCom-1.2  18 S
BayCom-     19
OpenBCM-    19
```

Damit ist sichergestellt, dass ab der BCM-Mailbox 1.30 und der OpenBCM auch der Forward von AutoBIN-Mails funktioniert.

Wird beim Forward ein CRC-Fehler erkannt, so wird dieser durch das Prompt "|>" statt ">" gekennzeichnet. Die Mail wird dann bis zu viermal wiederholt.

34.3. The CHECK command

Um Kompatibilität zu den DieBox-Systemen zu gewährleisten, wurde zusätzlich zum Befehl DIR NEWS der Befehl CHECK eingebaut. Dieser Befehl liest aus der Datei *check.bcm*, die chronologisch aufgebaut ist und dasselbe Format hat, wie die Board-Listen *list.bcm* in den jeweiligen Verzeichnissen der Boards.

Die Datei *check.bcm* wird neu erzeugt, wenn sie fehlt oder fehlerhaft ist. Dazu werden von allen Boards die Dateien *list.bcm* aneinandergereiht. Sind die Listen alle eingelesen, so wird die Datei lexikalisch sortiert. Dieser Vorgang kann nicht im RAM, sondern muss auf der Festplatte erfolgen, da die Datei sehr groß werden kann (bei 30000 Mails ca. 4 MB). Der Sortiervorgang kann deshalb sehr lange dauern (je nach Rechner/Datenbestand) und erfolgt nur, wenn es unbedingt notwendig ist. Ein absichtliches Erzeugen von *check.bcm* ist mit dem Befehl REORG C möglich.

Bei der Ausgabe des Befehls CHECK wird vom Ende zum Anfang hin gelistet. Da zum Zeitpunkt der Auflistung die Nummer einer Mail innerhalb eines Boards nicht bekannt ist, wird die Datei *checknum.bcm* angelegt. In dieser Datei ist zu jeder in *check.bcm* aufgelisteten Mail die Position der Mail innerhalb ihres Boards abgespeichert.

34.4. Multitasking structure

Über die Mailboxsoftware wird ein Task-Scheduler gelegt, der folgende Eigenschaften hat:

- Dynamisches Erzeugen und Löschen von Tasks. Eine Task kann an jeder beliebigen Stelle im Programm eröffnet werden (beliebige Schachtelungstiefe). Eine Task wird wieder gelöscht, wenn die zugehörige Prozedur beendet ist oder ein kill-Aufruf erfolgt (z.B. bei Disconnect).
- Unterbrechung des Ablaufs nur auf Anforderung (non-preemptive scheduling), d.h. der laufende Code wird nie an unvorhersehbaren Stellen unterbrochen. Dadurch braucht die gesamte Software nicht konsequent reentrant zu sein. Ausnahme: Bei allen Ausgaben muss mit einer Unterbrechung des Ablaufs gerechnet werden.
- Task-Kontrollblock: Alle vom Kontext abhängigen globalen Variablen werden im TCB untergebracht. Zu diesem wird ein stets gültiger Pointer bereitgehalten, über den alle TCB-Variablen referiert werden.

34.5. Regular expressions

Regular expressions are signs used for searching purposes. They are used in OpenBCM in files *convert.bcm*, *reject.bcm* and most search processes (commands CHECK, DIR, LIST...).

```
Sign     Description
 ^       finds beginning of a line at the beginning of a string
```

```
 $       finds end of line at the end of string
 .       finds any sign
 *       after string, finds every search string followed with any sign (or no
         sign).
         Example: "bo*" finds "bot", "bo" and "boo" etc but not "b"
 +       after string finds every search string followed with any sign but
         no further signs.
         Example: "bo+" finds "boo" and "booo", but not "bo" and "be"
 \       means to use following sign as search string
         Example: "\^" finds "^" and didn't search for beginning of line
 [ ]     finds every single sign
         Example: [bot] finds b, o or t
 [^]     this means negation
         Example: [^bot] finds all signs but no b, o or t
 [-]     means a rangs of letters
         Example: [b-o] finds every letter between b and o
```

The use of curved brackets (means "{" or "}") can't be used in OpenBCM, although in other implementations of regular expressions they are common.

Same examples:

```
 ^[WE][WU]$     finds as search string "WW", "WU", "EW" or "EU".
 ^S\:           finds "S:" only at beginning of a string.
```

35. FAQ – frequently asked questions

F: Ist ein Umstieg von FBB/DieBox/MSYS auf OpenBCM möglich?
A: Für FBB und MSYS gibt es keine Utilities zum Konvertieren der Datenbestände. Hier ist es am Besten, die Daten über ein Kabel (KISS oder AXIP) von der "alten" Box auf die "neue" Box zu forwarden, falls das die "alte" Software zulässt. Für DieBox gibt es sowohl eine Software zum Konvertieren der Mails als auch eine Möglichkeit zu Übernahme der Benutzerdaten:

- Mit dem Programm BOX2BCM von OE6BUD können die Mails konvertiert werden. Mails, die einen AutoBIN-Teil enthalten, werden dabei allerdings nicht konvertiert. BOX2BCM generiert aus den Einträgen im INFO-Verzeichnis die neuen Dateien für die OpenBCM-Mailbox. Das Programm BOX2BCM muss dazu im Boot-Verzeichnis stehen, ebenso müssen die Dateien LIFETIME.BOX und CONFIG.BOX der DieBox noch vorhanden sein. Der Pfad für das Info-Verzeichnis wird aus der Datei CONFIG.BOX gelesen. Die Lifetime für einen Eintrag wird aus LIFETIME.BOX gelesen. Achtung: Bei DieBox bedeutet die Lebensdauer "0", dass die Mail nie gelöscht wird, während das bei der OpenBCM-Mailbox bedeutet, dass die Mail beim nächsten Purge gelöscht wird. Für eine unendliche Lebensdauer muss bei der OpenBCM-Mailbox "999" angegeben werden. Da das Programm schon relativ alt ist, kann es zu Fehlern beim Import kommen.
- Die Benutzerdaten aus der DieBox-Datei *user3.idx* können von der OpenBCM-Mailbox direkt importiert werden (Befehl UIMPORT).

F: Bei der DOS-Version wird nach einigen Minuten der Monitor dunkel - ist etwas defekt?
A: Nein, das ist der Bildschirmschoner, er kann mit dem Befehl "crtsave 0" abgeschaltet werden.

F: Bei der Mailbox DB0XYZ läuft schon eine neue Version, woher kann ich diese Version bekommen?
A: Neue Versionen werden in das Board BAYBOX mit dem Verteiler BAYCOM eingespielt. Sollte die neue Version noch nicht dort zu finden sein, so handelt es sich um eine Beta-Version, also um eine Testversion, welche bei einigen wenigen Boxen gestestet wird, bevor sie verteilt wird. Eventuell lohnt sich auch ein Blick nach http://dnx274.dyndns.org/baybox

F: Wenn ein Run-Utility Bulletins löscht, so werden Fernerase-Mails erzeugt. Kann ich etwas dagegen tun?
A: Ja, mit einem Trick: Die Mails nicht löschen, sondern deren Lifetime auf #0 setzen. So werden die Bulletins genauso wie bei ERASE unsichtbar und beim nächsten Purge gelöscht. Man sollte sich als Sysop jedoch darüber im Klaren sein, dass man dann diese Mails tatsächlich löscht, bevor sie ggf. dem Forward-Partner angeboten wurden – man filtert somit im Netz und löscht nicht nur in der eigenen Box!!!

F: Warum wird nach dem Hochstarten einer Box mehr freier Speicher beim Befehl VERSION dargestellt, als wenn die Box eine Weile läuft (DOS)?
A: Die Speicherverwaltung der BayCom-Mailbox nutzt die malloc() und free() Aufrufe der C-Laufzeitbibliothek. Diese verwaltet den Speicher jeweils zu Blockgrößen, wie sie angefordert wurde. D.h. wenn zuerst ein sehr großer Block angefordert wird und dann ein sehr kleiner, und dann der große Block wieder freigegeben wird, entsteht eine Lücke. Der angezeigte freie Speicher ist aber stets nur der größte, zusammenhängende Block, und nicht die Summe aller "Schnipsel". Bleibt eine Lücke frei, so kann diese bei der nächsten Anforderung wieder genutzt werden, weil die Verwaltung stets nach dem ersten freien Block sucht, in den der angeforderte Block hineinpasst. Dabei entsteht natürlich ein Verschnitt, weil viele der angeforderten Blöcke eine unterschiedliche Größe haben. So erklärt sich, warum der angezeigte freie Speicher mit der Zeit immer weniger wird: Der Speicher wird immer mehr fragmentiert. Wird ein Block allerdings wieder freigegeben und sind in der Nachbarschaft ebenfalls freie Blöcke, so werden diese wieder zusammengefügt und es entsteht wieder durchgehender, freier Speicher. Der Vorgang der Fragmentierung setzt sich deshalb nicht fort, sondern pegelt sich auf einen durchschnittlichen Wert ein, der dann auf Dauer im Schnitt erhalten bleibt. Manche Blöcke bleiben allerdings sehr lange belegt (z.B. Sprachen, *convert.bcm*, *fwd.bcm* und solche Listen), weshalb auch der freie Speicher lange unverändert zu sein scheint. Üblich ist, dass gegenüber dem maximal möglich verfügbaren Speicher nach dem Hochstarten etwa 30-50 kB abgehen. Dies ist vor allem beim Start von externen DOS-Programmen lästig, da für diese nur durchgehend freier Speicher nutzbar ist.

F: Untersucht man die Datei *bcm.exe*, kommen Befehlswörter und Meldungen zutage, die keine oder eine unbekannte Wirkung haben. Was hat es damit auf sich und warum sind diese nicht dokumentiert?
A: Es gibt in den Quellen zur Box viele Baustellen, an denen irgendeine Funktion angedacht war aber nie realisiert wurde, oder die Teile werden auch zu anderen Zwecken verwendet und erfüllen ihre Funktionalität in der Box nur teilweise. In jedem Fall sollten nur die Funktionen als vorhanden gewertet werden, die auch dokumentiert sind. Da beim Erstellen der Boxsoftware eine relativ penible Versionsverwaltung betrieben wird und alle Erweiterungen automatisch in die Änderungslisten einfließen, gibt es kaum undokumentierte Funktionen, die nicht bewusst undokumentiert geblieben sind.

F: Im Syslog tauchen immer wieder völlig unverständliche Meldungen auf. Wie ist zu unterscheiden, welche Meldungen wirklich wichtig sind und was ignoriert werden kann, ggf. welche Gegenmaßnahmen notwendig werden.
A: Das ist ein sehr wunder Punkt. Zum einen ist klar, dass eine Dokumentation eigentlich dringend nötig wäre, zum anderen gibt es aber so furchtbar viele Meldungen (ca. 400 verschiedne), die potentiell kommen können, dass es ein gewaltiger Aufwand wäre sie alle zu dokumentieren. Prinzipiell kann gesagt werden, dass die Kategorie "#L" meist völlig harmlos ist, "#S" ist je nach Laune des Entwicklers entweder schlimm oder nicht, hingegen "#F" und "#A" sind stets höchst unangenehm und führen grundsätzlich dazu, dass das gewünschte Ergebnis nicht erzielbar ist. Eine Liste oft vorkommender Meldungen wird baldmöglichst nachgereicht.

F: Eingegebene Nachrichten werden sehr schnell an andere Boxen weitergegeben. Dies nimmt die Möglichkeit, nochmals über den Inhalt einer abgeschickten Nachricht nachzudenken und diese ggf. noch vor Weitergabe zu löschen.
A: Die unverzügliche Weitergabe von Nachrichten zu den Nachbarboxen war ein wichtiges Kriterium bei der Entwicklung der OpenBCM-Mailbox. Eine Nachricht sollte stets so schnell wie möglich am Ziel ankommen, sofern dies die Linkstrecken ermöglichen. Es ist deshalb nicht vorgesehen und im momentanen Design auch gar nicht ohne weiteres machbar, Nachrichten verzögert loszusenden. Eine Lösung für Kurzentschlossene ist, vor Abschluss der Nachricht deren Inhalt nochmals zu prüfen und diese ggf. durch CTRL-X oder /AB zu verwerfen und somit gar nicht erst abzuschicken.

F: Unter ALTER REJECT wäre es wünschenswert, mehr Begriffe anzugeben als in die momentan verfügbare 80-Zeichen-Zeile passen.
A: Dieser vielfach geäußerte Wunsch ist angekommen, ist jedoch nur mit sehr viel Ungemach zu verwirklichen. In der derzeitigen User-Struktur sind dafür nur 80 Zeichen vorgesehen und ein Umorganisieren dieser Struktur würde einen enormen Aufwand für alle Boxbetreiber beim Update bedeuten. Es wäre zum einen sehr viel freier Plattenplatz (ca. 100 MB) erforderlich, zum anderen wäre ein Zurückrüsten auf alte Versionen nicht mehr möglich. Außerdem würde das reorganisieren der Daten vor dem Update ca. 1 Stunde die Box lahm legen. Bei der Vielzahl der laufenden Boxen lohnt es sich durchaus, für einen solchen Schritt mehrere Änderungen auf einmal zusammenzufassen und erst dann eine solche Umorganisation durchzuführen.

F: Es war einmal eine 32 Bit-DOS-Version der OpenBCM-Mailbox angedacht und wurde auch angekündigt. Was ist daraus geworden?
A: Es gab eine 32 Bit-DPMI-Version von *bcm.exe*. Diese hat sehr viel Mühe bei der Portierung bereitet, da viele unter DOS mögliche Zugriffe im protected mode ganz anders sind, erst recht wenn diese von einem 32 Bit-Programm aus erfolgen. Auf den ersten Blick sieht dann alles auch ganz gut aus, man kann frei Speicher adressieren, es existiert eine virtuelle Speicherverwaltung, die sehr großzügige Speicherzugriffe ermöglicht. Außerdem entfällt die modulweise Segmentierung, was den Code angenehm übersichtlich macht. Es hat sich jedoch gezeigt, dass auch ganz krasse Nachteile entstehen, die das Projekt letztlich dann nach dessen Fertigstellung zu Fall gebracht haben:

- Zugriffe auf AX.25-Treiber (in diesem Fall *l2.exe*) sind extrem langsam, da bei jedem einzelnen Zugriff sehr viel im DPMI-Kern abläuft.
- Auch DOS-Zugriffe jeglicher Art sind ausgesprochen langsam, aus gleichem Grund.
- Hardware-Interrupts werden sehr häufig verschluckt, da auch diese vom DPMI-Kern geregelt werden müssen und während Umschaltvorgängen oft für lange Zeit gesperrt sind. Dadurch verliert *l2.exe* sehr häufig Pakete.
- Eine Zeitsynchronisierung mittels Timer-Interrupt ist sehr schwierig und anfällig, insbesondere weil es nicht gelingt, die Serviceroutine vom Memory-Paging auszuschließen und dadurch die Stabilität in Frage steht.
- Ansteuerung der Hardware ist nicht ohne weiteres möglich, dadurch keine serielle Schnittstelle und kein Watchdog etc. Auch der Bildschirm ist sehr langsam, weil intern aufwendiger.
- Insgesamt läuft die Box im 32 Bit-Mode um etwa einen Faktor 2 langsamer.
- Die DPMI-Bibliothek von Borland ist als Speicherfresser zu betrachten. *bcm32.exe* läuft nicht ohne weiteres auf einem mit 4 MB (!) bestückten Rechner, zumindest bleibt dann keinerlei Platz mehr für Dateisystem-Puffer. Das unterste MB wird faktisch hergeschenkt.

Obige Argumente machen klar, warum das Projekt nach näheren Tests zu den Akten gelegt wurde. Schade um die Arbeit und die dadurch entstandenen Kosten. Speziell bei letzterem ist ein besonderer Dank an DL6FBS zu richten, der hier den wesentlichen Teil übernommen hat.
In der Zwischenzeit gibt jedoch es eine Version der OpenBCM-Mailbox für Windows.

F: In der PS-Liste stehen oft nicht nur eingegebene Kommandos, sondern auch Symbole, die offenbar vom Programm generiert werden. Was bedeuten diese?
A: Beim Forwarding werden hier normalerweise die übertragenen Mails bzw. die ausgetauschten Kommandos dargestellt. Wird gerade keine Mail gesendet oder empfangen, so ist im Kommandoteil der PS-Liste der Zustand der Forward-Routine zu entnehmen. Das bedeutet im Einzelnen:

fwd_loop
Die Forward-Hauptschleife wurde aufgerufen

fwd_send
Die Forward-Schleife verzweigt demnächst zu einem SEND-Befehl

fwd_delay
Es wurden Nachrichten gesendet und es wird abgewartet, ob noch eine Mail für den Partner eintrifft bevor die Verbindung beendet wird

fwd_rec
Die Forward-Maschine erwartet ein Kommando vom Partner

fwd_pro
Es wurde ein Prompt gesendet

fwd_connect
Ein Connect-Befehl wurde abgesetzt, Verbindung wird erwartet

fwd_connected
Die AX.25-Verbindung ist gelungen

wait_for_sid
Ein Forward-Partner wurde connected, und es wird nun die SID des Partners erwartet

wait_pr1
Es wird der erste Prompt nach dem Login erwartet

wait_pr2
Es wird der zweite Prompt nach dem Senden der SID erwartet

rx_wait_sid
Ein Forward-Connect wurde entgegengenommen und die eigene SID gesendet; es wird nun die SID des Partners erwartet

F: Was bedeutet "NO - BID" beim Forward?
A: Das ist kein Fehler (also kein BID-Mangel o. ä.), sondern zeigt an, dass die angebotene Bulletin bereits lokal vorhanden ist und deshalb vom Partner nicht geschickt werden soll.

F: Darf die Baycom-Mailbox im CB-Funk benutzt werden?
A: Die OpenBCM-Mailbox ist für den CB-Funk freigegeben. Es handelt sich hierbei um eine mündliche Vereinbarung mit dem Autor, um einen Ansturm an Support-Fragen zu vermeiden. Fragen aus dem CB-Bereich zu Einstellungen, Installation, Bugreports, Programmierung und Sonstigem deshalb bitte an Jonas, DO1MJJ/DJJ812 oder per Email an "djj812@dbo812.de"/ PR-Mail: "DJJ812@DBO812.#M.OBB.BAY.DEU.EU" richten. Ebenso ist die Internetseite http://www.dbo812.de die offizielle Homepage für die CB-Version der OpenBCM-Mailbox.

F: Wie kann ich die OpenBCM-Mailbox für CB-Funk umstellen?
A: Der Rufzeichensatz für CB-Rufzeichen wird mit dem Befehl CALLFORMAT und evtl. auch noch LOGINCALLFORMAT aktiviert, wobei ersterer für nur für den Forward und der zweite für die Userlogins gilt. Dieser muss vor dem 1. Start der OpenBCM-Mailbox am besten manuell in die *init.bcm* eingetragen werden. Die Position spielt dabei keine Rolle. Als Parameter wird eine Ziffer verwendet:

0 = Amateurfunkrufzeichen
1 = CB-Rufzeichen
2 = Beide Rufzeichengruppen

Für CB-Funk muss also "CALLFORMAT 1" oder "CALLFORMAT 2" eingetragen werden. Ist CALLFORMAT auf 1 oder 2 gesetzt, wird auch "(CB)" als Hinweis für Aktivierung der CB-Version hinter jeder Mailbox-Versionsangabe mit angezeigt.

F: Welche Rufzeichen darf ich im CB-Funk verwenden?
A: Es besteht im CB-Funk zwar keine Rufzeichenpflicht, allerdings dürfen keine bereits vergebenen Rufzeichen, sei es angemeldete CB-Rufzeichen, Amateurfunkrufzeichen oder Rufzeichen irgendeines anderen Funkdienstes, verwendet werden! Generell wird die Anmeldung einer Mailbox auf CB empfohlen, um eine gewisse Organisation zu ermöglichen. Mehr hierzu auf der Homepage der Bundesnetzagentur (ehemals RegTP), unter http://www.bundesnetzagentur.de.

F: Beim Versuch in den Filesurf zu kommen, bekomme ich immer die Meldung "filesurf not enabled".
A: Der Filesurf ist nur fester Bestandteil der Linux- und Windows-Version. Er muss erst mit dem Sysopbefehl "fspath" aktiviert werden, wobei der angegebene Pfad vorhanden und für die Userrechte der OpenBCM freigeschaltet sein muss. Für DOS ist ein externes Programm wie z.B. EL besser, da hier der Arbeitsspeicher so knapp ist, dass die Filesurf-Option meist nicht mit einkompiliert wird.

F: Warum funktioniert der eingebaute Texteditor "edit", der Rejecteditor "REJEDIT", der Convert-Editor "CONVEDIT" oder der Forward-Editor "FWDEDIT" nicht?
A: Die eingebauten Editoren sind in der DOS-Version häufig nicht mit in die Mailbox einkompiliert, da sie einiges von dem raren Hauptspeicher fressen. Unter Linux/Windows sind diese Editoren jedoch normalerweise immer verfügbar.

F: Kann ein User die Packetlänge für sich einstellen?
A: Der Sysop kann für jeden User mit Hilfe des SETUSER-Befehls die Paketlänge (PACLEN) einstellen, dies gilt auch für die Forward-Partner. Der Eintrag in der "ALTER"-Liste erscheint nur, wenn die Packetlänge größer als 0 ist.
Die globale Paketlänge kann mit "PACLEN" eingestellt werden. Der Wert darf zwischen 40 und 255 liegen, bei dem Wert 0 wird die maximale Länge verwendet.

F: Wie aktiviere ich den Autorouter?
A: Der Befehl dazu heißt "autofwdtime". Er bestimmt, wie alt eine eingetroffene Nachricht in Tagen sein darf, um aus ihr den Pfad auslesen zu dürfen. Ist der Wert 0, ist der Autorouter deaktiviert. Zusätzlich muss der Befehl "afwdlist" in die Datei *crontab.bcm* eingetragen werden und dort mindestens einmal täglich durchlaufen werden.

F: Wie hebe ich das Download-Limit auf?
A: Der Befehl dazu heißt "setuser status 1". Die Größe des Limits kann mit dem Befehl "userquota" in KB angegeben werden.

F: Arbeitet die BCM32 mit Flex32 zusammen?
A: Ja! Dazu muss in der *init.l2* nur folgender Eintrag gemacht werden:

```
assign axip
peer 127.0.0.1
port 4722
txport 4721
```

Bei Flex32 muss im Control-Center zusätzlich ein AXIP-UDP-Port erstellt werden, der dann mit Port 4721 empfängt und mit Port 4722 sendet.

F: Arbeitet die BCM32 mit XNET(NT) zusammen?
A: Ja! Dazu muss in der *init.l2* nur folgender Eintrag gemacht werden:

```
assign axip
peer 127.0.0.1
port 4724
txport 4723
```

Bei XNET muss in der *autoexec.net* ein AXIP-UDP-Port hinzugefügt werden, der dann mit Port 4723 empfängt und mit Port 4724 sendet. Dies ist beispielsweise ab Version 1.30 von XNET mit folgender Zeile machbar:

```
att ip0 axudp 0 1 l4723 d4724 127.0.0.1
```

F: Wie bekomme ich die OpenBCM unter Windows9x ans laufen?
A: Unter Windows9x (95, 98, Millenium) kann nur die DOS-Version der OpenBCM-Mailbox verwendet werden. Diese läuft bisher nur mit Flexnet für DOS oder Flex95

zusammen. Hier muss allerdings versucht werden, konventionellen Speicher, also unterhalb der 640KB-Grenze, freizubekommen. Ein Betrieb mit Flex32 oder dem Flexnet DOS-Node ist unter Windows 9x nicht möglich.

F: Läuft die DOS-OpenBCM unter Windows9x mit XNET zusammen?
A: Die OpenBCM für DOS benötigt Flexnet als Treiber. Es ist möglich, Flex95 (übrigens auch Flex32) mit XNET(NT) zu verbinden. Die funktioniert mit AXIP-UDP-Ports. In XNET(NT) kommt folgender Eintrag in die *autoexec.net*:

```
att ip0 axudp 0 1 l4726 d93 127.0.0.1
```

Hierzu muss Flex95 VOR Windows mit folgenden Parametern gestartet werden:

```
flexnet
ether32 /c=1 /n0=AXIP_UDP,127.0.0.1:4726
flex
tfemu (nur notwendig wenn auch ein DOS-Terminal mitlaufen soll)
```

Bei Flex95 müssen die Einstellungen im Control-Center vorgenommen werden. Flex32 wird von der DOS-OpenBCM nicht unterstützt.
Achtung: Werden weitere AXIP-UDP-Ports in XNET benutzt, dürfen diese nicht auf dem UDP-Port 93 empfangen, da Flex95 diesen bereits belegt und nicht geändert werden kann!

F: Läuft die OpenBCM für DOS mit Flex32 unter Windows?
A: Ein Betrieb mit Flex32 oder dem Flexnet DOS-Node ist unter Windows 9x nicht möglich. Nur der Betrieb der OpenBCM für Windows ist mit Flex32 oder XNET möglich.

F: Wie unterdrücke ich den Forward von 7PLUS und AUTOBIN-Dateien?
A: Einfach beim jeweiligen Forward-Partner in der *fwd.bcm* die Option "-K", "-L", "-M" oder "-N" angeben.

F: Die Zeitzone wird in der OpenBCM nicht richtig angezeigt, warum arbeitet die OpenBCM überhaupt mit unterschiedlichen Zeiteinstellungen?!
A: Mit dem Befehl "time -a" lässt sich die Systemzeit und die Differenz zur UTC (Welt)-Zeit anzeigen. Diese Einstellung muss unbedingt stimmen, damit u. a. der Autorouter richtig funktioniert.
Unter Windows kam es hier vermehrt in alten Versionen zu Problemen, insbesondere dann, wenn unter der Systemsteuerung unter "System", "Erweitert" eine Umgebungsvariable "TZ" falsch definiert wurde. Am besten ist, man löscht diese Umgebungsvariable ganz und verlässt sich nur auf die Systemeinstellungen des Betriebsystems (die natürlich richtig eingestellt sein sollten). Falls man die Variable unbedingt nutzen muss, weil z.B. ein anderes Programm diese Variable zwingend benötigt, sollte sie in Westeuropa möglichst auf "MET+1MEZT+2" eingestellt sein. Die OpenBCM verfügt über eine automatische Sommer-/Winterzeit-Umstellung, sollte dies nicht gewünscht sein, so ist die Variable "summertime" in *init.bcm* auf "0" zu setzen, ansonsten sollte die Variable auf "1" stehen.

F: Das Telefonmodem reagiert nicht in der DOS-Version.
A: Der IRQ muss mit der Schnittstelle übereinstimmen. Das Echo des Modems muss in einem normalen Terminalprogramm unter derselben Schnittstellengeschwindigkeit wie in der OpenBCM mit ATE0 ausgeschaltet werden und die Einstellung mit ATW gespeichert werden.

F: Die OpenBCM beendet sich ständig selbst.
A: Das liegt vermutlich an zu alten MSG-Dateien. Ein Hinweis darauf ist die Meldung "missing lines" in der *trace/syslog.bcm*. Das Format der Dateien hat sich ab der BCM v1.43 geändert und alte Sprachdateien sind immer noch im Umlauf. Durch das Austauschen mit aktuellen MSG-Dateien sollte das Problem behoben sein. In den aktuellen Versionen kann man mit "a s" auch die Versionsnummer der MSG-Dateien abfragen. Sollte hier keine Versionsnummer stehen, sind die MSG-Files definitiv veraltet!

F: Wieso verlangt die Linux/Windows-OpenBCM ein Passwort bei Login via Telnet/HTTP?

A: Jeder Benutzer benötigt ein eigenes Passwort, um sich via HTTP oder Telnet einloggen zu können. Dies stellt der Sysop mit dem Befehl "setuser ttypwd" ein. Es darf max. 8 Zeichen lang sein.

F: Änderungen an der Konfiguration werden nicht erkannt.
A: Damit die Änderungen erkannt werden, muss NEW eingegeben oder die OpenBCM neu gestartet werden. Beim Speichern der Konfigurationsdateien muss auf die Endung *.bcm* geachtet werden. Bis auf die *init.l2* haben alle Dateien diese Endung. Im Zweifelsfall verwendet einen anständigen Texteditor (unter Windows NICHT Wordpad - empfehlenswert ist z.B. Textpad, siehe http://www.textpad.com) der auch CR/LF am Zeilenende beherrscht.

F: Woher kommen Übertragunsfehler / CRC-Fehler?
A: Meistens treten diese auf einem KISS-Link auf. Um dies zu verhindern muss der CRC-Modus aktiviert werden. Dies geschieht bei Flexnet mit der Option "c" hinter der Baudrate mit "fset" und bei XNET durch Angabe der Port-Mode-Parameters "2".

F: Wie schließe ich ein TNC2 an die OpenBCM an?
A: Ein TNC kann von der OpenBCM nur mit KISS direkt angesteuert werden, es ist jedoch davon ABZURATEN! Besser ist die Verwendung einer geeigneten Netzknotensoftware zwischen Mailbox und Außenwelt.

F: XNET(NT) und AXIP-Links (auch dynamische IPs) – wie geht das?
A: Bei festen IP-Adressen reicht ein Eintrag in der *autoexec.net*:

```
att ip0 axudp 0 1 l93 d93 192.168.1.1
```

"ip0" sowie "0" sind die jeweiligen IP- und XNET-Ports, l93 und d93 der Empfangs- und Sende-Port für AXIP-UDP-Packete und 192.168.1.1 die IP-Adresse des Linkpartners. Die angegebenen UDP-Ports müssen in einer evtl. vorhandenen Firewall freigeschalten sein. Möchte man über das Internet eine Node-Verbindung aufbauen, ist es in den meisten Fällen nicht möglich, da die Nodes über dynamische IP-Adressen verfügen. Damit diese Programme arbeiten können, benötigt man einen DNS-Eintrag, den man z. B. bei www.dyndns.org oder www.myip.org bekommen kann. Um den Abgleich der IP-Adressen zu diesen DNS-Einträgen zu automatisieren, gibt es wieder entsprechende Tools (DynDNS- Clients). Unter Windows hat sich das Programm "ynSite" dafür sehr bewährt. Um die IP-Adressen auch im entsprechenden AXIP-UDP-Treiber im XNET(NT) einzustellen, bedarf es eines weiteren Tools, das zyklisch gestartet werden muß und anhand der DNS-Einträge die neuen IPs einstellt, z.B. NTNETDNS (auf der Downloadseite von http://www.dbo812.de zu finden. Im Archiv befindet sich auch eine Installationsanleitung dazu.

F: Flex95 und AXIP-Links (auch dynamische IPs) – wie geht das?
A: Notwendig sind dafür nur die entsprechenden Treiber in *autoexec.bat* (bevor Windows gestartet wird). Der AXIP-Port wird unter DOS mit dem Treiber IPPD realisiert. Hier sind keine dynamischen IPs möglich und auch nur LAN-interne AXIP-Links. Unter Windows geht es mit dem Treiber ETHER32, allerdings sieht es damit besser aus.

- Unter DOS:
  Für IPPD wird vor dem Laden von Flexnet noch ein sog. Packet-Treiber für die Netzwerkkarte benötigt, der meistens im Lieferumfang bei den Treiberdisketten dabei ist. Hierbei handelt es sich meistens um eine kleine, speicherresidente *.com*-Datei, der als Parameter meistens die IO-Adresse, der Hardware-IRQ und der Software-IRQ, den man sich aussuchen kann, der aber meistens 0x60 ist, angegeben werden muss. Unter Windows muss das TCP/IP-Protokoll installiert sein. Folgende Batchdatei kann verwendet werden:

  ```
  ne2000 0x300 10 0x60
  flexnet
  ippd -c:1 -m:192.168.1.1 -p:192.168.1.2 -u:93 -i0x60
  flex
  fset digicall MYCALL
  tfemu
  ```

192.168.1.1 ist im Beispiel die IP-Adresse des Rechners auf dem die Batchdatei gestartet wird, 192.168.1.2 ist die des Nachbarrechners. 93 ist der UDP-Port und 0x60 die Nummer des Software-Interrupt des verwendeten Packet-Treibers. Die Angabe von c=1 bestimmt die Anzahl der AXIP-Ports in Flexnet, alle weiteren Ports können mit dem Zusatzprogramm IPPDCFG eingestellt werden.

- Unter Windows:
  Für ETHER32 braucht man weitaus weniger Angaben, da die eigentlichen Netzwerkaufgaben Windows übernimmt. Folgende Batchdatei kann verwendet werden:

```
flexnet
ether32 /c=1 /n0:AXIP_UDP,192.168.1.2
flex
fset digicall MYCALL
tfemu
```

  192.168.1.1 ist im Beispiel die IP-Adresse des Rechners auf dem die Batchdatei gestartet wird, 192.168.1.2 ist die des Nachbarrechners. 93 ist der UDP-Port. Bei ETHER32 ist nur die Angabe der Nachbar-IP notwendig, da er die Netzeinstellungen von Windows benutzt. Die Angabe von c=1 bestimmt die Anzahl der AXIP-Ports in Flexnet, alle weiteren Ports können mit dem Zusatzprogramm ETH32CFG eingestellt werden. Will man nun über das Digicall linken und man hat mehrere Ports installiert, muss man als SSID nur die Portnummer angeben. z. B. MYCALL-1 um über Port 1 zu linken.

Bei dynamischen IP-Adressen empfiehlt sich die Verwendung von DNS-Einträgen, hierzu kann man unter Windows/Flex95 das Programm ETH32DNS benutzen.

F: Ständig kommt beim Senden von Nachrichten "Befehl oder Dateiname nicht gefunden".
A: In der DOS-Version 1.43 bis 1.44 der BCM war dies ein Fehler der BCM und trat meist nur auf, wenn sehr wenig Arbeitspeicher vorhanden war. Nachfolgende Versionen, so auch die OpenBCM, haben diesen Fehler nicht mehr. Es wird also Zeit für ein Update!

F: Wie lege ich Forward-Zeiten in der OpenBCM fest?
A: Jeder Eintrag in der *fwd.bcm* beginnt mit der Kopfzeile, die das Boxcall, die Zeitzeile und den Connect-Pfad enthält, z.B.:

```
CB0BOX AAAAAAAAAAAAAAAAAAAAAAAP CB0NOD / CB0BOX CB0DIG
 THEBOX $WP
```

Die Zeitzeile besteht dabei aus den Stunden des Tages, jeweils ein Buchstabe für eine Stunde beginnend mit 0 Uhr bis 23 Uhr. Soll also z.B. um 1 Uhr Nachts der Forward gestartet werden muss der zweite Buchstabe ein "A" sein.
A = Forward von Info & Usermails
U = Forward von Usermails
P = Postforward
. = kein Forward
Weitere Informationen zur Forwarddatei sind im Kapitel "9. Complete forward configuration" ausführlich beschrieben!

F: Beim Start der OpenBCM kommt die Meldung "Windows 95 or Win32s is not recommended for bcm32!"
A: Die Windows-Version der OpenBCM ist nur für Windows-Versionen mit NT-Kern geeignet. Also nur Windows NT 4.0, Windows 2000 und Windows XP (und folgende). Windows 95, 98, und ME sind nicht geeignet. Dort kann nur die DOS-Version mit Flex95 als Interface benutzt werden.

F: Bei Verwendung der Windows-Version der OpenBCM unter Windows XP kann ich keine Mails speichern, im Trace sieht man die Fehlermeldung:
"#F savemail: fopen […] errno=2 No such file or directory" - ist das ein Fehler der Software, denn unter Windows NT oder 2000 tritt dies nicht auf?

A: Unter Windows XP ist es wohl nötig die Pfade in der Datei *init.bcm* komplett anzugeben, also z.B. "userpath c:/bcm/user/". Unter Windows NT/2000 reicht die Angabe von "userpath user".

F: Ich sehe im Syslog immer nur z.B. folgende Ausgabe "#L savemail: m_filter '/usr/bin/m_filter.prg' not found!" - woran liegt das?
A: Ist das M_Filter <Programm> nicht vorhanden, oder ist der Pfad falsch eingestellt, kommt es beim Senden von Mails zu einer solchen Ausgabe im Syslog. Ferner könnte unter Linux auch noch die Möglichkeit bestehen, dass die Dateizugriffsberechtigungen des M_Filter-Programmes falsch gesetzt sind (es muss mindestens 'rx' gesetzt sein). Um das Mailfilterprogramm ganz zu deaktivieren, muss nur die Option M_FILTER OFF in der Datei *init.bcm* gesetzt werden.

F: Ich sehe im Taskmanager immer einen Connect der eigenen Mailbox und die Auslastung des Systems steht dann die ganze Zeit auf 100%. Die Festplatte rappelt. Der Zustand dauert scheinbar ewig. Ist die Mailbox abgeschmiert?
A: Vermutlich steht z.B. im Connecttext das "checkcount"-Makro. Connectet dann z.B. ein Forward-Partner und hat der noch nie einen Mail"Check" im System ausgeführt, so zählt das "checkcount"-Makro alle "neuen" Mails zusammen. Bei einer größeren Mailbox mit vielen Rubrikmails kann das dann schon mal länger dauern... als Abstellmaßnahme sollte man also das "checkcount"-Makro nicht im Connecttext der Mailbox oder bei "A C" global verwenden!

F: Kann man Mails automatisch in eine Forward-Datei exportieren?
A: Klar geht das - dazu ist nur eine Zeile in *crontab.bcm* mit dem Befehl einzufügen, z.B.

```
fwdexp db0xyz fwd/export/db0xyz.exp
```

Sind für mehrere Mailboxen Forwarddateien zu erstellen, so ist es im Allgemeinen übersichtlicher, eine Datei *mailexport.imp* mit den jeweiligen Export-Zeilen anzulegen und dann nur diese eine Datei über *crontab.bcm* mit z.B.

```
20 6,18 * * * mailexport
```

aufzurufen.
Es ist ferner zu beachten, dass die gemailte Forward-Datei anschließend gelöscht wird, bevor die Mailbox einen erneuten Mailexport-Aufruf macht - sonst bekommt der Forwardpartner die gleichen Mails mehrfach!

F: Wenn man um 15:00 Uhr eine Mail in der OpenBCM verfasst, und diese Mail dann wieder ausliest, wird die Uhrzeit 13:00 Uhr anstatt 15:00 Uhr angezeigt, das ist doch nicht richtig oder?
A: Doch, das ist alles korrekt! In allen Mailboxsystemen wird mit Weltzeit (auch UTC oder GMT Zeitzone genannt) gearbeitet. Die deutsche Sommerzeit unterscheidet sich hiervon um plus 2 Stunden, die deutsche Winterzeit um plus 1 Stunde. Wenn man den Befehl "time -a" in der OpenBCM eingibt, bekommt man sowohl die lokal eingestellte Uhrzeit, als auch die Weltzeit angezeigt.

F: Ich bekomme den automatischen HTTP Zugang für Gäste einfach nicht zum Laufen. Habe folgende Einstellungen:

```
HTTPACCOUNT 1
HTTPGUESTFIRST 1
GUESTCALL GAST
```

Sobald aber jemand die Mailbox per HTTP "connected", wird er immer noch nach dem Benutzernamen und Passwort gefragt. An was könnte das liegen?
A: Für den User "GAST" ist vermutlich kein TTYPW gesetzt. Um dies zu überprüfen, einfach mal als Sysop "A GAST" abfragen und den Zahlenwert bei TTYPWlen überprüfen. Ist er auf "0", so ist kein Passwort eingestellt. Um dann ein Passwort zu setzen, reicht ein "setuser gast ttypw xyz123" aus. Da HTTPGUESTFIRST auf 1 gesetzt wurde, wird dann der GAST-User jedes Mal automatisch ohne Passwortabfrage eingeloggt.

F: Ich habe einen Forward von meiner Mailbox DB0ABC-3 zu einer anderen Mailbox DB0XYZ-8 meines Erachtens richtig eingerichtet, jedoch kommt ein Mailaustausch immer noch nicht zustande! Im Trace-Fenster sehe ich, dass die andere Mailbox DB0XYZ-8 nach einem Verbindungsausbau immer "*** no logintime" sendet und dann die Verbindung wieder kappt:

```
12:fm DB0ABC-3 to DB0XYZ-8 ctl I00^ pid F0 [29]
[OpenBCM-1.03-AB1D1FHMRW$]
>
12:fm DB0XYZ-8 to DB0ABC-3 ctl RR1v
12:fm DB0XYZ-8 to DB0ABC-3 ctl I10^ pid F0 [255]
*** no logintime
```

Bei einem Connectversuch von DB0ABC-3 nach DB0XYZ-8 passiert etwas ähnliches:

```
12:fm DB0XYZ-8 to DB0ABC-3 ctl I15^ pid F0 [40]
[OpenBCM-1.03-AB1D1FHMRW$] 1406031330
>
12:fm DB0ABC-3 to DB0XYZ-8 ctl RR6v
12:fm DB0ABC-3 to DB0XYZ-8 ctl I61^ pid F0 [25]
*** no password expected
```

Woran liegt das, und wie kann ich den Forward richtig einrichten?
A: Bei der Partnermailbox DB0XYZ-8 (ebenfalls ein OpenBCM-System) ist offensichtlich ein Diebox-Passwort für den Forward eingerichtet (Datei *db0abc.pwd*). Vielleicht war die eigene Mailbox DB0ABC-3 vorher schon mal unter einem Diebox-System ein Linkpartner von DB0XYZ-8. Wenn man bei der Mailbox DB0XYZ-8 die Datei *db0abc.pwd* löscht oder verschiebt, funktioniert der Forward, allerdings ist dieser dann nicht mehr passwortgeschützt. Alternativ könnte man auch bei der Mailbox DB0ABC-3 die gleiche Datei, aber unter dem Namen *db0abc.pwd* anlegen. Noch besser wäre es jedoch, bei beiden Systemen auf die PWD-Dateien zu verzichten und für den Forward das MD5-Passwortverfahren anzuwenden, d.h. in beiden Mailboxen für das jeweils andere Mailboxrufzeichen mit "SETUSER <call> SF MD5" das MD5 Verfahren aktivieren, um dann mit "SETUSER <call> PW <passwort>" das entsprechende (gleiche) Passwort zu setzen.

F: Bei STATUS FORWARD wird ja eine Statistik zum Forward angezeigt. Offensichtlich hat der Autor bei den Statusmeldungen einen Tippfehler eingebaut, ab und an erscheint dort "recv" und mal "recvp" bzw. "send" und mal "sendp".
A: Nein! Das ist alles korrekt! Bei "recv" wird tatsächlich gerade eine Mail empfangen, bei "recvp" findet ein receive-proposal (=Empfangsvorschlag) mit der anderen Mailbox statt. Analog gilt dies auch für "send" und "sendp". Alle Statusmeldungen sind übrigens bei HELP STATUS FORWARD genau erklärt.

F: Beim Forward zwischen DIEBOX und OpenBCM werden bei Usermails die MIDs nicht übergeben, woran liegt das?
A: DIEBOX ab v1.9c1 kann auch MIDs verwalten, daher funktioniert dies auch prinzipiell. Bei DIEBOX gibt es aber eine Datei *mbsys\sfwid.box* in der für BCM und OpenBCM folgende Einträge zu machen sind:

```
BayCom-1.1  18 S
BayCom-1.2  18 S
BayCom-     19
OpenBCM-    19
```

Fehlt die letzte Zeile, dann passiert das in der Frage beschriebene Verhalten.

F: Ich will einen Forward-Connect via Telnet machen. Es geht aber kein Connect raus, was mache ich bloß falsch?
A: Evtl. ist in der *fwd.bcm* die Zeile für den Connect-Pfad falsch eingestellt. Ein üblicher Fehler ist etwa die Zeile mit

```
DBO123 AAAAAAAAAAAAAAAAAAAAAAAP telnet://dbo123.dyndns.org:4719
```

anstelle von

```
DBO123 AAAAAAAAAAAAAAAAAAAAAAAP telnet:dbo123.dyndns.org:4719
```

abzugeben (das :// ist durch ein : zu ersetzen!).

F: Ich nutze OpenBCM als User-Mailbox. Wenn ich eine Mail eingebe wird diese auch wie gewünscht sofort zur Heimatmailbox gesendet. Neue Mails in der Heimatmailbox werden jedoch nur dann abgeholt, wenn ich selber eine Mail dorthin schicke. Normalerweise sollte die doch durch den Forward-Eintrag in *crontab.bcm*

```
...
3,33      *    *    *    *      forward
....
```

halbstündlich erfolgen, zumal im SLOG folgender Hinweis kommt:

```
startfwd: checking DB0CZ
```

Was geht hier bloß schief?
A: Vermutlich ist in der Forwarddatei *fwd.bcm* folgender Eintrag zu finden:

```
DB0CZ AAAAAAAAAAAAAAAAAAAAAAAA DG1GGG / DB0CZ
```

Damit der Forward auch tatsächlich nach DB0CZ angestoßen wird, ist statt "A" ein "P" einzugeben, also

```
DB0CZ PPPPPPPPPPPPPPPPPPPPPP DG1GGG / DB0CZ
```

Der Eintrag im slog sagt bei "A" nur aus, dass überprüft wird, ob Mails für DB0CZ vorliegen, nicht aber, dass der Forward auf alle Fälle aufgebaut wird. Dies wird erst mit "P" gemacht.

F: In meinem SYSLOG sehe ich beim Senden einer Mail massenweise Zeilen, wie z.B.:

```
14.12.03 15:44:50l DGT274: #L parse_rej: invalid rej-action:
14.12.03 15:44:50l DGT274: #L parse_rej: invalid rej-action:
14.12.03 15:44:50l DGT274: #L parse_rej: invalid rej-action:
14.12.03 15:44:50l DGT274: #L parse_rej: invalid rej-action:
```

Die Datei *reject.bcm* habe ich überprüft, die Einträge sind aber alle richtig. Woran liegt das?
A: Es liegt an der *reject.bcm*! Dort sind mit Sicherheit Zeilen enthalten, die mit einem Leerzeichen beginnen, und das ist nicht erlaubt! Sollen Kommentarzeilen eingefügt werden, so müssen diese mit einem Semikolon beginnen.

F: Mein Forward mit einer FBB-Mailbox funktioniert nicht. Im Trace-File sieht man z.B.:

a)

```
28.01.04 04:47:06l 15R FA P MS3BOX DOK346 WP 11764-MS3BOX 80
28.01.04 04:47:06l 15R FA P MS3BOX DOK346 WP 11786-MS3BOX 193
28.01.04 04:47:06l 15R FA P MS3BOX DOK346 WP 11800-MS3BOX 188
28.01.04 04:47:06l 15R FA P MS3BOX DOK346 WP 11831-MS3BOX 355
28.01.04 04:47:06l 15R FA P MS3BOX DOK346 WP 11876-MS3BOX 184
28.01.04 04:47:06l 15R F> 00
28.01.04 04:47:06l 15S FS +++++
28.01.04 04:47:06l 15R ---- [mail header]
28.01.04 04:47:26l 15R ---- [mail body]
28.01.04 04:47:26l 15S *** rcvd invalid block-id
28.01.04 04:47:26l 15R ---- [yapp error]
28.01.04 04:47:26l 15S *** FBB forwarding error
28.01.04 04:47:27l 15- ---- logout
```

b)

```
08.08.04 19:33:23z 13R *** connected to AP1BOX
08.08.04 19:33:23z 13R [FBB-7.00i-AB1FHMRX$]
08.08.04 19:33:23z 13R C_FILTER 2000 7.0f3 DOR-Edition by DOR693
08.08.04 19:33:23z 13R Forward !
08.08.04 19:33:23z 13R FWD OK BBS-Oelde
08.08.04 19:33:23z 13R
08.08.04 19:33:23z 13R Forward Port der AP1BOX,
08.08.04 19:33:23z 13R
08.08.04 19:33:23z 13R Es liegen hier f□r Dich 4 Mails mit 4 KB vor.
08.08.04 19:33:23z 13R
08.08.04 19:33:23z 13R (1) AP1BOX >
08.08.04 19:33:23z 13S [OpenBCM-1.06b24-AB1FHMR$]
08.08.04 19:33:23z 13- ---- [FBB Fwd outgoing connection]
08.08.04 19:33:23z 13S FA P DGT274 AP1BOX.#NDS.DEU.EU APOLO1 88EDBO27401C 410
08.08.04 19:33:23z 13S FA P DGT274 AP1BOX.#NDS.DEU.EU APOLO1 88EDBO27401D 503
08.08.04 19:33:23z 13S FA P DGT274 AP1BOX.#NDS.DEU.EU APOLO1 88EDBO27401F 439
08.08.04 19:33:23z 13S FA B HF1BKM DL MEINUN 8E8NB1BKM_04 893
08.08.04 19:33:23z 13S FA P DGT274 AP1BOX.#NDS.DEU.EU PING 88EDBO27401H 216
08.08.04 19:33:23z 13S F> C5
08.08.04 19:33:34z 13R *** Checksum error (Checsum of proposals is wrong.)
08.08.04 19:33:34z 13S *** received status prompt invalid
08.08.04 19:33:34z 13S *** FBB forwarding error
08.08.04 19:33:35z 13- ---- logout
```

A: Das liegt definitiv an FBB bzw. dessen (schlechter) Konfiguration. Anscheinend kann man die SID-Erkennung bei FBB komplett lahmlegen und stattdessen manuell die Optionen für einen Forwardpartner definieren. Und da kann man natürlich dann eine Menge in FBB VERKONFIGURIEREN! So kann man in der Forward-Datei *forward.sys* von FBB die unterstützten Forwardprotokolltypen des Forward-Partners fest eintragen (Parameter N). Definitiv geht der Forward nicht, wenn man dort die Option "X-Forward" aktiviert (also den Wert von N auf 8 oder größer einstellt). Im Zweifelsfall sollte man es mal mit "N 5" oder "N 1" probieren. In WinFBB kann man die Option X-Forward auch im Konfigurationsprogramm generell deaktivieren. Ggf. kann man in FBB aber auch die automatische SID-Erkennung wieder aktivieren (wenn man weiß ob und wo das denn

geht). Bei weiteren Problemen sollte man auch seltsame Connect-Filter-Programme, die zwischen dem Forward von OpenBCM und FBB laufen, deaktivieren!

F: Mein Forward mit einer FBB-Mailbox funktioniert nicht. Im Trace-File sieht man z.B.:

```
07.09.04 22:41:49l  2R *** connected to DBX645
07.09.04 22:41:49l  2R [FBB-7.04h-AB1FHMR$]
07.09.04 22:41:49l  2R DBX645 BBS forward port.
07.09.04 22:41:49l  2R
07.09.04 22:41:49l  2R (1) DBX645 BBS>
07.09.04 22:41:49l  2S [OpenBCM-1.06b25-AB1FHMR$]
07.09.04 22:41:49l  2- ---- [FBB Fwd outgoing connection]
07.09.04 22:41:49l  2S FA P DBO274 DBX396 MW1RBL 69EDBO27400L 36494
07.09.04 22:41:49l  2S FA P DBO274 DBX396 MW1RBL 69EDBO27400M 36494
07.09.04 22:41:49l  2S FA P DBO274 DBX396 MW1RBL 69EDBO27400N 36494
07.09.04 22:41:49l  2S FA P DBO274 DBX396 MW1RBL 69EDBO27400O 36494
07.09.04 22:41:49l  2S FA P DBO274 DBX396 MW1RBL 69EDBO27400P 36494
07.09.04 22:41:49l  2S F> 33
07.09.04 22:41:50l  2R FS NNA15250YY
07.09.04 22:41:50l  2S *** invalid fbbstatus character
07.09.04 22:41:50l  2S *** FBB forwarding error
07.09.04 22:41:52l  2- ---- logout
```

A: OpenBCM unterstützt erst seit v1.06b28 den alternativen Antwortbuchstaben "A" im Resume-Modus. Wenn man OpenBCM auf die aktuelle Version umrüstet, wird der Forward also funktionieren!

F: Ich habe bislang OpenBCM 1.03 unter Windows betrieben, und habe nun die EXE-Datei mit OpenBCM 1.05 oder höher ausgetauscht und natürlich die MSG-Dateien auch aktualisiert. Wenn sich nun jemand in die Mailbox einloggen will, kommt aber immer nur eine Abfrage des Namens - und das in einer Endlosschleife. Hilfe!
A: In OpenBCM 1.04 wurde das Format der Userdatenbank geändert. In 1.03 hieß die Datei unter Windows noch *users.bcm*, seit 1.04 heisst sie sowohl unter DOS/Windows als auch Linux *users4.bcm*. Vermutlich ist der erste Start der neuen Version (manuell) unterbrochen worden und die Konvertierung von *users.bcm* nach *users4.bcm* wurde nicht richtig durchgeführt. Das Result ist nun eine defekte *users4.bcm* Datei. Lösung: OpenBCM beenden, *users4.bcm* löschen und dann nochmal die neue OpenBCM-Version starten. Dann wird *users4.bcm* neu aus *users.bcm* erzeugt.

F: Wer kann mir kurz mitteilen, wie ich in der OpenBCM per Import BIN-Dateien einlese. Irgendwo stand das mal, ich finde es aber nicht wieder. Hintergrund: Ich will mir per *crontab.bcm*-Eintrag täglich eine Datei von der Platte als persönliche Mail schicken.
A: Der Befehl heißt ".attach <file>" und wird einfach in die zu sendende Nachricht am Ende eingefügt. Einfaches Beispiel:

```
s dh6bb daten
.attach platte/datei
```

Es wird kein ***end oder so etwas benötigt, da ja ein BIN-File kommt.

F: Ich habe ein (kleines) Problem mit unserer OpenBCM DB0WHV: Ein OM hatte in seiner Urlaubszeit seine MyBBS nach DB0WHV gesetzt. Jetzt steht sie wieder auf DB0SIF, aber die alten Mails liegen immer noch bei DB0WHV. Ein "postfwd" hilft da auch nicht. Es wird nichts in den Forward geleitet.
A: Nachrichten, die ihre Zielbox erreichen, bleiben auch nach Ändern der MyBBS-Information liegen. Dies ist kein Bug, sondern ein Feature. Es verhindert, dass Nachrichten durch MyBBS-Verstellereien auf die Reise geschickt werden können. Es muss explizit ein FORWARD-Befehl auf diese Nachrichten angesetzt werden.

F: Ein OM hat seine MyBBS bei einer (in meiner Box) unbekannten Box. Nach Eintragen dieser Box in die *fwd.bcm* bleibt aber auch diese Mail liegen. Wo liegt hier das Problem? Zwischen den Ohren?
A: Vermutlich haben diese Nachrichten ein HOLD-Flag gesetzt (siehe DIR -P), dann müssen sie mit "FORWARD -H <...> @ <...>" 'losgelöst' werden.

F: Ich möchte in *init.l2* gerne mehrere Ports definieren. Ein Port ist für die Linkanbindung und Forward zu benutzen, und zusätzlich möchte ich gerne zwei

AXUDP Links nutzen, die Links zu Sysop-PC's darstellen. Eingehende Connects werden auch von der Mailbox auf allen Ports beantwortet, aber es geht kein Forward-Connect nach draussen.
A: Die Reihenfolge in *init.l2* ist ausschlaggebend. Auf dem Port, der als erstes definiert ist, werden ausgehende Connects aufgebaut. Ggf. muss hier also umsortiert werden.

F: Jetzt ist es mir schon mehrfach passiert, dass die neuen Mails in den Rubriken nicht in mein NNTP News-Reader gelangen. Gerade kamen wieder ein paar Mails in der Rubrik ALLE an und ich wollte die mit dem NNTP-Reader auslesenm, aber sie wurden dort gar nicht angezeigt. Ich musste wieder alle Nachrichten zuruecksetzten und dann neu runterladen, damit ich die neuen Mails lesen konnte.
A: Genau das kann passieren, wenn zwischendurch Mails aus der Rubrik per PURGE gelöscht wurden, und dann anschließend weniger Mails als z.B. am Vortag in der Rubrik vorhanden sind! Hier hilft nur, im NNTP-Client, also z.B. Outlook, diese Newsgroup zurückzusetzen und dann die Rubrik komplett neu abzurufen! Das Problem tritt nicht auf, wenn die Mailbox ohne Rubriklöschung ("nopurge") läuft.

F: Ich habe eine Rubrik WINDOWS in meiner Mailbox angelegt. Nun schreibt jemand in den Nachbarboxen Mails in die Rubrik WIND, in meiner Box sind diese Mails nun plötzlich alle in der Rubrik WINDOWS zu finden, obwohl ich dies gar nicht in der *convert.bcm* eingetragen habe. Was läuft da schief, bzw. wie kann ich die Mailbox dazu bringen, bei mir auch die Mails in WIND anzuzeigen.
A: Durch die automatische Rubrikenkomplettierung werden Mails für die Rubrik WIND wie Mails für die Rubrik WINDOWS behandelt, da in deiner Mailbox noch keine Rubrik WIND existiert. Wenn du die Rubrik WIND anlegst (mkboard-Befehl), werden die Mails auch nicht mehr in der Rubrik WINDOWS landen.

F: Ich bekomme jeden Monat einen Report, wie z.B.:

```
>-------------------------------------------------------------------
>Unknown target bbs "p -sfn":
>-----------------------------------------------------------
>Mails Since Address                                  Forward
>    3 135d: DH4LAR.#DB0SYL.SLH.DEU.EU               -unknown-
>    0  15d: EW3DS.BLR.EU                            -unknown-
>    0 164d: WA8WNI.#SEOH.OH.USA                     -unknown-
>    4  51d: IZ6ASI                                  -unknown-
>
>-----------------------------------------------------------
```

>Wie bringe ich das in Ordnung?
A: Der monatliche Report soll dazu dienen, solange zu nerven, bis man das Routing in Ordnung bringt :-) - man trägt dazu die momentan nicht routbaren Boxen oder besser deren Verteiler in die *fwd.bcm* ein. Im obigen Beispiel also am besten den Regionalverteiler .SLH, den Landeskenner .BLR und die zwei Einzelrufzeichen IZ6ASI bzw. WA8WNI. Zu welchen Forwardpartner die Mails am sinnvollsten geschickt werden sollten, kann man meist mit z. B. "p -a <call>" herausbekommen - dort wird angezeigt, über welche Forwardpartner die Mails dieser Box eintrudeln, und wieviele Hops die Mails zurücklegen.

F: Ich finde zu einem neuen Befehl keine Infos in der Onlinehilfe (help-Befehl).
A: Die Onlinehilfe wird regelmässig aktualisiert. Vermutlich ist sie nicht auf dem aktuellen Stand - dann sollten zumindest *msg/help.dl* bzw. *msg/help.gb* aktualisiert werden. Mit dem Befehl "h hver" kann man den Versionsstand abfragen. Auf der Internetseite http://dnx274.dyndns.org/baybox/pre kann man sich immer den aktuellen Stand an msg-Dateien herunterladen.

36. Copyright

For OpenBCM mailbox the "GNU General Public License" Version 2.0, June 1991 takes effect.

37. Literature

[1] Sysop-Documentation FBB Version 7.00d, Jean Paul Roubelat, F6FBB@F6FBB.FMLR.FRA.EU, 1997

[2] Sysop-Manual THEBOX Version 1.9, Reinhard Rüdiger, DF3AV@DK0MAV.#NDS.DEU.EU, 1992

[3] Dokumentation zum BayCom-Node 1.54/1.55a, Florian Radlherr, DL8MBT@DB0AAB.#BAY.DEU.EU, 1994, 1995

[4] Manual BayCom-Terminal 1.50, Florian Radlherr, DL8MBT, 1992

[5] Manual BayCom-Terminal 1.60, Florian Radlherr, DL8MBT, 1994

[6] Sysop-Manual MSYS Version 1.19, Mike Pechura, WA8BXN, September 1995

[7] Manuals zur W0RLI-BBS Version 17.5, Henry N. Oredson, W0RLI@W0RLI.OR.USA.NOAM, 1994

[8] Update-Infos THEBOX Version 1.9b4-1.9b5, Reinhard Rüdiger, DF3AV, Juni 1994, Sommer 1995

[9] FlexNet 3.3g/h - Dokumentation, Gunter Jost DK7WJ@DB0ZDF, Juni 1995, 1996

[10] Digipoint 5.07 Dokumentation, Joachim Schurig, DL8HBS@DB0GR.#BLN.DEU.EU, 1997

[11] PC/FlexNet 3.3e - Dokumentationen zu den div. Programmteilen Gunter Jost, Januar 1996

[12] ISO 3166 2 und 3 stellige Ländercodes. 15.12.1993, ISO 3166:1993 (E/F) Maintanance Agency beim DIN Berlin, Tel. +49 30 26 28791-2861

[13] BBS Hierarchical Addressing Protocol x.3.4, TAPR Special Interest Group, Dave Wolf WO5H, März 1995

[14] WinGT für Amateurfunk, Gerd Mitländer, DG8NDL@DB0RT.#BAY.DEU.DEU, Version 1.60, April 1996

[15] Computer Networks, Andrew S. Tanenbaum, 3. Auflage, Amsterdam 1996

[16] RFC821 Internet-Übertragungsprotokoll für E-Mails

[17] RFC822 Internet-Nachrichtenformat für E-Mails

[18] RFC1866 Hypertext Markup Language 2.0

[19] RFC2068 Hypertext Transfer Protocol 1.1